PHP 网络编程技术与实例

四维科技　曹衍龙　赵斯思　编著

人 民 邮 电 出 版 社

图书在版编目（**CIP**）数据

PHP 网络编程技术与实例 / 四维科技编著. —北京：人民邮电出版社，2006.3

ISBN 7-115-14529-6

Ⅰ. P… Ⅱ. 四… Ⅲ. PHP 语言－程序设计 Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2006）第 009469 号

内 容 提 要

本书全面、细致地介绍了 PHP 网络开发技术的原理和基础编程知识，不仅包括 PHP 开发环境的配置、PHP 的基本语法等内容，还包括 PHP 的高级开发技术、数据库编程技术等内容。

全书共分为 3 篇 20 章，以“基础篇→高级开发篇→案例实战篇”为线索具体展开，涵盖了 PHP 网络编程的方方面面。而且，对于书中每个知识点的讲解，都配有大量可实际运行的实例，供读者边学习边实践，以便读者快速、全面地掌握 PHP 的开发方法和技巧。

本书内容由浅入深，从最基本的 PHP 语法到综合案例，比如在线图片管理系统、电子邮件收发系统和在线书店系统等，都有详细的讲解。随书配套光盘中提供了书中涉及实例的全部源代码，以方便读者举一反三，编写出适合自己的程序。

本书不仅适合初学者学习 PHP 网络开发技术，还能够帮助有一定编程经验的 PHP 开发人员解决开发过程中遇到的问题。本书可作为高等院校相关专业的教材和参考用书，也可作为 PHP 的培训用书，或广大用户的自学用书。

PHP 网络编程技术与实例

- 编　　著　四维科技　曹衍龙　赵斯思
 责任编辑　汤　倩
- 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
 邮编　100061　　电子函件　315@ptpress.com.cn
 网址　http://www.ptpress.com.cn
 北京鸿佳印刷厂印刷
 新华书店总店北京发行所经销
- 开本：787×1092　1/16
 印张：34.25
 字数：827 千字　　2006 年 3 月第 1 版
 印数：1 – 5 000 册　　2006 年 3 月北京第 1 次印刷

ISBN 7-115-14529-6/TP・5251

定价：56.00 元（附光盘）

读者服务热线：(010)67132692　印装质量热线：(010)67129223

前　言 *Foreword*

随着 Internet 的高速发展，网络规模的迅速扩大，各种各样的网站层出不穷，ASP、JSP 和 PHP 等服务器端脚本语言逐渐成为了网站开发的主流。尤其是以 PHP 技术为基础的黄金组合 Apache+MySQL+PHP 的广泛应用，使得电子商务类网站更是发展迅速。

- 为什么写本书

虽然已出版的关于 PHP 的书籍较多，但大都没有将基础知识和应用实例进行有机的结合，要么忽略了最基本的 PHP 开发环境的配置和基本语法的讲解，要么缺少对综合实例及复杂系统的全面介绍，给读者的学习带来不便。

本书汇集了作者几年来从事 PHP 网络开发工作的经验之精华，全面、系统地介绍了 PHP 网络开发的基本原理和编程方法，并通过大量实例展示了 PHP 项目的开发步骤和编程技巧。目的就是帮助读者真正学到实用的内容。

- 本书特色

✓ 内容全面。全书共分为 3 篇 20 章，详细地介绍了 PHP 开发技术的各个方面，方便读者快速地找到自己感兴趣的和对自己有帮助的内容。

✓ 实例丰富。书中几乎所有的知识点都对应有可实际运行的实例，方便读者一边学习编程原理，一边进行上机实践，从而加深对书中的重点、难点内容的理解。

✓ 提供了模块化的案例。书中最后的“案例实战篇”，提供了常用模块的编程实现，以及 8 个综合性的案例，方便读者根据自己的需求，进行代码的复用，以便节省开发时间。

✓ 光盘中提供了每个实例的完整源代码，方便读者举一反三，开发出适合自己的工程项目。

- 技术支持

本书主要由曹衍龙编写，参加写作的人员还有赵斯思、汪杰、茅健和赵玉玺等。在编写过程中，我们力求精益求精，但难免存在一些不足之处。如果读者使用本书时发现差错或者遇到问题，敬请批评指正，并请发 E-mail 到 forway@zj.com 和 tangqian@ptpress.cn 与我们联系。

编　者

目　录

Contents

第一篇　基　础　篇

第 1 章　黄金组合 Apache+MySQL+PHP 简介 …… 2

1.1　PHP 开发语言 …… 2

1.2　Apache 服务器 …… 3

1.3　MySQL 数据库 …… 4

1.4　Apache+MySQL+PHP 环境配置 …… 4

1.4.1　Apache 的安装配置 …… 4

1.4.2　PHP 的安装配置 …… 11

1.4.3　MySQL 的安装配置 …… 14

第 2 章　PHP 语言轻松入门 …… 25

2.1　PHP 语言基础 …… 25

2.1.1　PHP 文件格式 …… 25

2.1.2　PHP 标记 …… 26

2.1.3　PHP 文件的访问形式 …… 28

2.1.4　PHP 程序注释 …… 29

2.1.5　PHP 语法概述 …… 30

2.2　PHP 的变量 …… 31

2.2.1　PHP 的变量命名 …… 31

2.2.2　PHP 的数据类型 …… 31

2.2.3　数据类型转换 …… 42

2.2.4　PHP 中的预定义变量 …… 45

2.2.5　变量的引用 …… 47

2.2.6　变量的变量 …… 48

2.2.7　常量 …… 49

2.3　PHP 的运算符 …… 51

2.3.1　算术运算符 …… 51

2.3.2　赋值运算符 …… 52

2.3.3　位逻辑运算符 …… 53

2.3.4　三元运算符 …… 54

2.3.5　比较运算符 …… 54

2.3.6　字符串运算符 …… 55

2.3.7　递增和递减运算符 …… 56

2.3.8 逻辑运算符 …… 57
2.3.9 运算符优先级 …… 58
2.4 表达式 …… 58
2.5 控制语句 …… 59
2.5.1 条件语句 …… 59
2.5.2 循环语句 …… 64
2.5.3 break 和 continue 语句 …… 67
2.6 数组 …… 69
2.6.1 数组类型 …… 69
2.6.2 数组初始化 …… 71
2.6.3 数组的应用 …… 72
2.7 函数 …… 87
2.7.1 内置函数 …… 88
2.7.2 自定义函数 …… 95
2.7.3 变量函数 …… 99
2.7.4 函数变量的作用域 …… 100
2.8 日期和时间处理 …… 103
2.8.1 获取日期和时间 …… 103
2.8.2 使用 getdate 函数获得日期信息 …… 105
2.8.3 使用 mktime 函数取得一个日期的时间戳 …… 106

第 3 章 字符串和正则表达式 …… 107

3.1 格式化字符串 …… 107
3.1.1 去除多余空格 …… 107
3.1.2 格式化字符串显示 …… 108
3.2 字符串的连接和分割 …… 115
3.3 字符串的比较 …… 118
3.4 字符串的查找和替换 …… 119
3.4.1 查找字符串 …… 119
3.4.2 定位字符串 …… 121
3.4.3 字符串替换 …… 123
3.5 正则表达式 …… 125
3.5.1 正则表达式基础 …… 126
3.5.2 匹配正则表达式模式 …… 126
3.5.3 锚定一个匹配 …… 128
3.5.4 匹配单个字符 …… 129
3.5.5 匹配字符类 …… 130
3.5.6 替换匹配 …… 131
3.5.7 不区分大小写的模式 …… 131
3.5.8 分组模式 …… 131

3.5.9　预定义字符和字符类 …… 132
3.5.10　限定符 …… 132
3.5.11　逆向引用 …… 133
3.5.12　其他常用正则函数 …… 133

第 4 章　PHP 与 HTML 表单 …… 137

4.1　HTML 表单 …… 137
4.1.1　GET 提交方法 …… 140
4.1.2　POST 提交方法 …… 142
4.2　获取提交的数据 …… 143
4.2.1　文本框 …… 144
4.2.2　单选按钮 …… 144
4.2.3　复选框 …… 144
4.2.4　列表框 …… 148
4.2.5　隐藏字段 …… 150
4.3　常用表单数据验证 …… 152
4.3.1　姓名验证 …… 152
4.3.2　电子邮件验证 …… 153
4.3.3　密码字段验证 …… 154
4.3.4　生日字段验证 …… 155
4.4　URL 编码解码函数 …… 156
4.4.1　urlencode 函数 …… 156
4.4.2　urldecode 函数 …… 157

第二篇　高级技术篇

第 5 章　PHP 中的文件处理技术 …… 160

5.1　检测文件或者目录是否存在 …… 160
5.2　打开文件 …… 161
5.3　写入或者追加数据 …… 163
5.3.1　写入数据 …… 163
5.3.2　追加数据 …… 165
5.4　关闭文件 …… 166
5.5　读取数据 …… 167
5.5.1　读取一行数据 …… 167
5.5.2　读取整个文件 …… 170
5.5.3　读取一个字符 …… 173
5.5.4　读取任意长度函数 …… 174
5.5.5　文件定位 …… 175

5.6 其他常用文件处理函数 …… 177
第 6 章 PHP 与面向对象程序设计 …… 184
6.1 面向对象编程知识点回顾 …… 184
6.1.1 对象 …… 184
6.1.2 类 …… 185
6.1.3 继承 …… 185
6.1.4 多态性 …… 186
6.2 在 PHP 中使用类 …… 186
6.2.1 创建类 …… 186
6.2.2 构造函数 …… 188
6.2.3 析构函数 …… 188
6.2.4 类的实例化 …… 188
6.2.5 使用类的属性 …… 189
6.2.6 类的访问控制 …… 190
6.2.7 类方法的调用 …… 191
6.2.8 类的应用实例 …… 192
6.3 面向对象编程的高级技术 …… 193
6.3.1 继承的实现 …… 193
6.3.2 多态的实现 …… 194
6.3.3 作用域分辨运算符 …… 196
6.3.4 parent 关键字 …… 197
6.3.5 final 关键字 …… 198
6.3.6 static 关键字 …… 199
第 7 章 错误调试与异常处理 …… 201
7.1 错误类型 …… 201
7.1.1 语法错误 …… 201
7.1.2 运行错误 …… 205
7.1.3 逻辑错误 …… 207
7.1.4 设置错误报告级别 …… 208
7.2 错误调试 …… 209
7.2.1 调试过程 …… 209
7.2.2 常用的调试手段 …… 210
第 8 章 MySQL 数据库 …… 211
8.1 MySQL 应用基础 …… 211
8.1.1 登录到数据库 …… 211
8.1.2 修改用户密码 …… 212
8.1.3 MySQL 的权限管理 …… 213
8.1.4 创建新用户 …… 214

8.1.5　创建数据库 …… 216
8.1.6　创建数据库表 …… 217
8.1.7　数据类型 …… 221
8.1.8　查看数据库 …… 225
8.1.9　创建索引 …… 228
8.1.10　备份数据库 …… 228
8.1.11　恢复数据库 …… 230
8.2　使用 MySQL 数据库 …… 230
8.2.1　插入数据 …… 230
8.2.2　查询数据 …… 233
8.2.3　编辑记录 …… 242
8.2.4　删除记录 …… 244
8.3　MySQL 的高级应用 …… 246
8.3.1　事务 …… 246
8.3.2　存储过程 …… 248
8.4　使用 MySQL Administrator 管理数据库 …… 252
8.5　使用 phpMyAdmin …… 257

第 9 章　PHP 的数据库编程 …… 258

9.1　通过 PHP 操作 MySQL 数据库 …… 258
9.1.1　PHP 连接 MySQL …… 258
9.1.2　选择数据库 …… 260
9.1.3　创建查询 …… 260
9.1.4　显示查询结果 …… 262
9.1.5　插入新数据 …… 268
9.1.6　编辑数据 …… 269
9.1.7　删除数据 …… 270
9.1.8　获取被影响的记录行数 …… 271
9.1.9　创建数据库 …… 272
9.1.10　创建数据库表 …… 272
9.1.11　事务处理 …… 273
9.1.12　存储过程 …… 275
9.1.13　其他重要函数 …… 276
9.2　mysqli 函数库应用 …… 282
9.3　通过 PHP 连接 SQL Server 数据库 …… 286

第 10 章　PHP 中的图像处理技术 …… 291

10.1　GD2 函数库中的常用函数 …… 292
10.2　利用 GD2 函数库进行图像处理 …… 297
10.2.1　创建图像 …… 297

10.2.2 图像处理的各种实例 …… 299

第 11 章 会话控制 …… 316

11.1 Cookie 的应用 …… 316
11.1.1 Cookie 的概念 …… 316
11.1.2 PHP 中的 Cookie 编程 …… 317
11.2 实现会话控制 …… 323
11.2.1 会话的基本方式 …… 323
11.2.2 创建会话 …… 323
11.2.3 会话应用实例 …… 324
11.2.4 会话控制的配置 …… 326

第三篇 案例实战篇

第 12 章 常用模块编程 …… 330

12.1 分页显示模块 …… 330
12.2 在线数据编辑模块 …… 334
12.3 文件上载模块 …… 342
12.4 聊天室模块 …… 346
12.4.1 用户登录界面（index.php） …… 346
12.4.2 登录成功界面（login.php） …… 347
12.4.3 聊天室的主界面（frames.php） …… 349
12.4.4 显示聊天内容界面（outputchat.php） …… 350
12.4.5 显示在线用户列表界面（outputusers.php） …… 351
12.4.6 输入聊天内容界面（input.php） …… 351

第 13 章 注册登录系统 …… 353

13.1 系统整体设计 …… 353
13.2 数据库安装配置 …… 353
13.3 系统的实现 …… 355
13.3.1 系统注册模块 …… 355
13.3.2 账号激活模块 …… 359
13.3.3 系统登录模块 …… 361
13.3.4 取回密码模块 …… 365
13.3.5 修改密码模块 …… 366
13.3.6 注销登录模块 …… 368
13.4 验证码的实现 …… 368
13.4.1 将验证码发送的客户端（index.php） …… 369
13.4.2 verificate.php …… 371

13.4.3 doimg.php …… 371
第 14 章 投票系统 …… 373
14.1 基于数据库的投票系统 …… 373
14.1.1 系统整体设计 …… 373
14.1.2 数据库设计 …… 374
14.1.3 系统实现 …… 374
14.2 基于文件的投票系统 …… 387
14.2.1 系统整体设计 …… 387
14.2.2 数据的存储形式 …… 387
14.2.3 系统实现 …… 388
第 15 章 计数器系统 …… 396
15.1 基于文本的图形显示计数器 …… 396
15.2 面向对象的图形显示计数器 …… 397
15.2.1 counter.php …… 398
15.2.2 counter.inc …… 398
15.2.3 databases.inc …… 401
15.2.4 实例运行演示 …… 401
15.3 综合计数器 …… 402
15.3.1 配置数据库 …… 402
15.3.2 显示日历及统计结果（counter.php） …… 402
15.3.3 连接数据库（dbconnection.php） …… 407
15.3.4 导航文件（top.php） …… 408
15.3.5 实例运行演示 …… 409
第 16 章 文件管理系统 …… 410
16.1 系统整体设计 …… 410
16.2 系统实现 …… 410
16.2.1 浏览目录和文件的基本信息 …… 410
16.2.2 删除文件 …… 417
16.2.3 编辑文件 …… 419
16.2.4 保存文件 …… 420
16.2.5 通过浏览器浏览文件 …… 420
16.2.6 创建目录 …… 421
16.2.7 上载文件 …… 422
16.2.8 删除目录 …… 422
16.2.9 设置权限 …… 422
第 17 章 在线图片管理系统 …… 424
17.1 系统整体设计 …… 424

17.2 创建数据库 424
17.3 系统实现 425
17.3.1 实现导航页面 425
17.3.2 设置系统参数 426
17.3.3 添加图片 427
17.3.4 查看图片 429
17.3.5 编辑图片 436
17.3.6 删除图片 437
第 18 章 功能完善的 BBS 系统 439
18.1 系统整体设计 439
18.2 数据库设计 440
18.2.1 数据库表设计 440
18.2.2 数据库安装配置 442
18.3 系统实现 447
18.3.1 系统注册/登录 447
18.3.2 发表文章 452
18.3.3 查看文章列表 455
18.3.4 查看文章 458
18.3.5 回复文章 464
18.3.6 查看个人信息 465
18.3.7 实现管理员功能 465
18.3.8 导航文件（index.php） 472
第 19 章 电子邮件收发系统 473
19.1 Mail 函数的简单应用 473
19.2 系统整体设计 475
19.3 系统实现 476
19.3.1 系统设置文件 config.php 477
19.3.2 POP3 系统登录 478
19.3.3 收取邮件和附件 481
19.3.4 阅读邮件和保存附件 487
19.3.5 发送或者回复邮件 489
19.3.6 创建文件夹 503
19.3.7 通讯录管理 503
19.3.8 系统设置 503
第 20 章 在线书店系统 504
20.1 数据库设计 504
20.2 在线书店的客户端功能 506
20.2.1 通用文件（Common.php） 507

20.2.2　数据库连接类（db_mysql.inc）…… 510
20.2.3　主界面文件（Default.php）…… 515
20.2.4　会员登录文件（Login.php）…… 517
20.2.5　显示图书详细信息（BookDetail.php）…… 522
20.2.6　购物篮管理（ShoppingCart.php）…… 522
20.2.7　修改购物信息（ShoppingCartRecord.php）…… 523
20.2.8　图书查询（Books.php）…… 523
20.3　在线书店的管理功能…… 524
20.3.1　会员管理…… 524
20.3.2　订单管理…… 525
20.3.3　图书管理…… 525
20.3.4　分类管理…… 526

第一篇

基础篇

- 黄金组合 Apache+MySQL+PHP 简介
- PHP 语言轻松入门
- 字符串和正则表达式
- PHP 与 HTML 表单

第 1 章　黄金组合
Apache+MySQL+PHP 简介

有人说 PHP 没有了跟 Apache 和数据库的结合，它的魅力就降低了一半，其实何止降低一半，没有了对数据库的支持，PHP 就基本没有了优势，而数据库中又以 MySQL 与 PHP 的结合最易上手，使用最灵活，功能最强大。

MySQL 是一款小巧的数据库服务器软件，对于小型（当然也不一定很小）应用系统是非常理想的。它由一个服务守护进程和很多客户端程序以及大量的库组成。除了支持标准的 ANSI SQL 语句，MySQL 还支持多种平台，而在 UNIX 系统上该软件支持多线程运行方式，从而能获得相当好的性能。它也可以在 Windows 平台上以系统服务方式或普通进程方式运行。

PHP 是一种服务器端解释的脚本语言。如果读者接触过 ASP 的话，那么对于在 HTML 页面中嵌入代码应该是比较熟悉了。PHP 代码在服务器一端被解释成普通的 HTML 页面内容，送给浏览器一端。这种模式使得我们可以用它来完成相当复杂的功能。

1.1　PHP 开发语言

下面介绍 PHP 的一些特点。

（1）学习起来快，容易上手。PHP 的学习过程非常简单。只要了解一些基本的语法和语言特色，哪怕不是 PHP 的语法，而是 C 语言或者 perl 的语法，就可以开始 PHP 编程之旅了。只要有编程基础，就完全可以边学 PHP 边做网站。

（2）数据库连接。PHP 与 MySQL 是绝佳的组合。但是 PHP 也可以编译成具有与许多数据库相连接的函数。PHP 还支持 Informix、Oracle、Sybase、Solid 和 PostgreSQL，以及通用的 ODBC。PHPLIB 就是最常用的可以提供一般事务需要的一系列基库。

（3）可扩展性。对于一个非程序员来说，为 PHP 扩展附加功能可能会比较难，但是对于一个 PHP 程序员来说并不困难。PHP 能发展到今天，本身就是大量程序员协同努力的结果，因此文档及邮件列表等支持方式很多。

（4）面向对象编程。PHP 提供了类和对象。PHP 支持构造器、提取类等。

（5）可伸缩性。传统上，网页的交互作用是通过 CGI 来实现的。CGI 程序的伸缩性不很理想，因为要为每一个正在运行的 CGI 程序开一个独立进程。解决方法就是将编写 CGI 程序的语言的解释器编译进 Web 服务器（比如 JSP）。PHP 就可以以这种方式安装。内嵌的 PHP 可以具有更高的可伸缩性。

PHP 5 于 2004 年 7 月 13 日正式发布。由于 PHP 在 Web 应用程序市场中的地位，因此该版本获得了广泛的重视。

PHP 5 中最基本、最重要的变化是使用了对象句柄（或 Id），而不是将其实现为自带数据

类型。复制时实际上只复制句柄（Id 编号）本身，并不复制这些句柄所代表的对象。

下面列出了 PHP 5 中的一些语言特性。

（1）对象克隆语义。无论是对象赋值、按值传递对象，还是从函数中按值返回对象，脚本引擎均不会自动克隆 PHP 5 中的对象。如果需要克隆，则可以使用 clone 关键字（例如“clone $obj;”）显式克隆对象。开发人员还可以在类中实现一个名为__clone()的方法，当克隆操作复制了所有原始对象的属性后，将在得到的新对象中调用此方法。

（2）公共/私有/受保护的访问修饰符。PHP 5 支持其他面向对象的语言，如 C++和 Java 中通常都具有的 PPP（公共/私有/受保护的）访问修饰符。可以将这些访问修饰符用于属性和方法，以施加访问限制。

（3）接口、抽象类和方法。PHP5 使用 Java 样式的接口和抽象类为多重继承（MI）提供解决方案。

（4）允许 PHP 扩展重载 PHP 对象语法。PHP 5 最重要的特性之一就是在对象语法和其语义之间存在一个抽象层。此方法允许 PHP 扩展创建它们自己的对象，这些对象的行为与用户级 PHP 对象的行为不同。

（5）PHP 5 中的 XML 和 Web 服务。XML 变得越来越重要，它允许不同的应用程序和系统使用标准工具和方法协同处理数据。这是 Oracle 及其他供应商坚定不移地支持和采用该技术的原因之一。

PHP5 除了对现有的 XML 扩展进行重新编写和统一以外，还出现了一个新的 XML 扩展。此扩展名为 SimpleXML，它允许开发人员能像访问自带 PHP 对象那样访问 XML 文件。

（6）SOAP 协议。使用 Web 服务（更具体地说是 SOAP 协议）解决两个或更多系统之间的互操作性问题已变得越来越普遍。PHP5 为 SOAP（客户端以及服务器 API）创建了一个新的自带实现，它允许 PHP 开发人员轻松创建和使用 Web 服务。

1.2　Apache 服务器

目前比较流行的 Web 服务器有 Apache 和 IIS，而相比较而言，Apache 服务器具有以下特点。

（1）Apache 能够支持多种 Web 编程语言，如 ASP、JSP 以及 PHP 等。

（2）由于采用了开放源代码的方法，因此 Apache 的安全性非常好。

（3）扩展性高。扩展性是指 WWW 服务提供工具是否可以应用于多种场合，多种网络情况，多种操作系统。Apache 是个多面手，它不仅仅应用于 Windows、对于 UNIX、Linux 以及 FREEBSD 等多种操作系统它都可以胜任工作。而且不同操作系统的配置步骤基本类似，可移植性非常高。

目前 Apache 的最新版本为 2.0，相比较以前的版本，Apache 2.0 增强了以下功能。

（1）UNIX 线程。在支持 POSIX 线程的 UNIX 系统上，现在 Apache 能在混合多进程、多线程模式下运行，使很多（但不是全部的）配置的可扩缩性得到改善。

（2）新的编译系统。重写了原来的编译系统，现在是基于 autoconf 和 libtool 的，使得 Apache 的配置系统与其他软件包更加相似。

（3）多协议支持。Apache 拥有能够支持多协议的底层构造。mod_echo 就是一个例子。

（4）对非 UNIX 平台更好的支持。Apache 2.0 在诸如 BeOS，OS/2 和 Windows 等非 UNIX

平台上有了更好的速度和稳定性。

（5）支持 IPv6。在所有能够由 Apache Portable Runtime 库提供 IPv6 支持的系统上，Apache 默认获得 IPv6 侦听套接字。

（6）过滤。Apache 的模块可以写成过滤器的形式，当内容流经它到服务器或从服务器到达的时候进行处理。

（7）多语种错误回报。返回给浏览器的错误信息用 SSI 文档实现了多语种化。管理员可以利用此功能进行定制以达到观感的一致。

关于 Apache 服务器的配置和使用参见 1.4 节。

1.3 MySQL 数据库

MySQL 是一款精巧的 SQL 数据库管理系统，以操作简便著称。

作为完全免费的数据库，MySQL 可以自由地用于个人和商业而不必支付任何费用。MySQL 是客户机/服务器模式的分布式系统，由 SQL 服务器、客户端、管理工具和应用编程接口等一整套工具组成。它的平台无关性优点使其能在多个操作系统上运行，其中包括 UNIX、Linux、Solaris、SunOS、FreeBSD 等。此外，MySQL 还是多线程的，不仅具有运行速度快，可处理大批量数据等特点，还支持多 CPU 使用。

MySQL 目前的最新版本为 5.0。虽然目前的数据库软件有很多，比如 SQL Server、Oracle、SyBase 等，但是同这些功能强大的商业数据库相比，MySQL 数据库具有以下的一些特点和优点。

（1）高效的执行效能。在各种专业的测试中，MySQL 在各方面的性能都不逊于那些商业的数据库软件。

（2）开源软件。这意味着用户可以免费得到，有利于该软件的普及以及发展。

（3）使用简单，安装非常方便。

（4）跨平台性。该系统能够在 Linux 和 Windows 等操作系统中运行。

关于 MySQL 数据库的安装配置，参见 1.4 节。

1.4 Apache+MySQL+PHP 环境配置

既然 Apache+MySQL+PHP 在网站开发方面有很大的优势，本书就主要围绕这 3 个软件进行介绍。在学习本书具体内容之前，首先要配置 PHP 的运行调试环境，下面介绍详细的配置步骤。

1.4.1 Apache 的安装配置

首先介绍 Web 服务器 Apache 2.0 的安装。该软件是免费软件，读者可以在网站“http://httpd.apache.org/download.cgi”或者“http://www.apache.com/en_apache/index.php”下载该软件。

下载界面如图 1-1 所示。

笔者使用的操作系统平台为 Windows XP，因此可以下载“Win32 Source”或者“Win32 Binary（MSI Installer）”。其中“Win32 Source”是源代码，需要读者自己编译执行，而“Win32

Binary（MSI Installer）”是已经做好的安装程序，建议读者直接下载该文件。

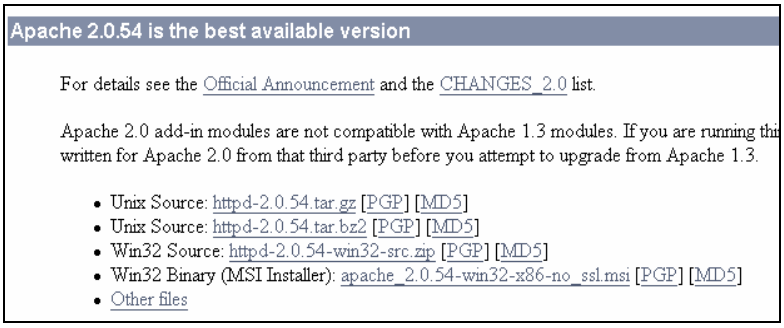


图 1-1　Apache 2.0 服务器下载界面

文件下载后，文件名为“apache_2.0.54-win32-x86-no_ssl.msi”，双击该文件可以进行安装，启动安装界面如图 1-2 所示。

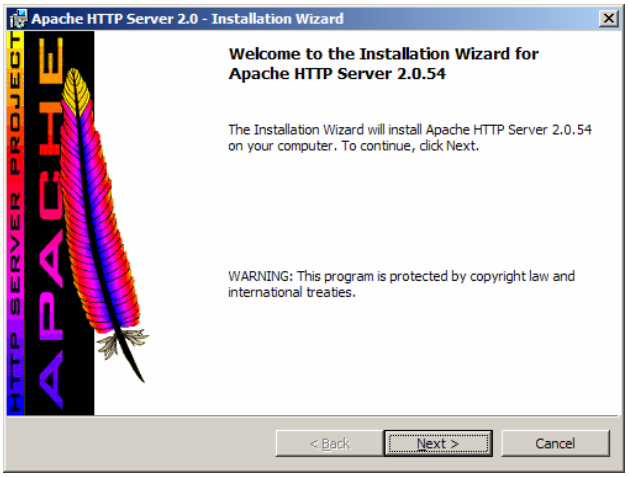


图 1-2　Apache 服务器安装向导

在图 1-2 中，单击“Next”按钮，进入到下一个安装界面，如图 1-3 所示。

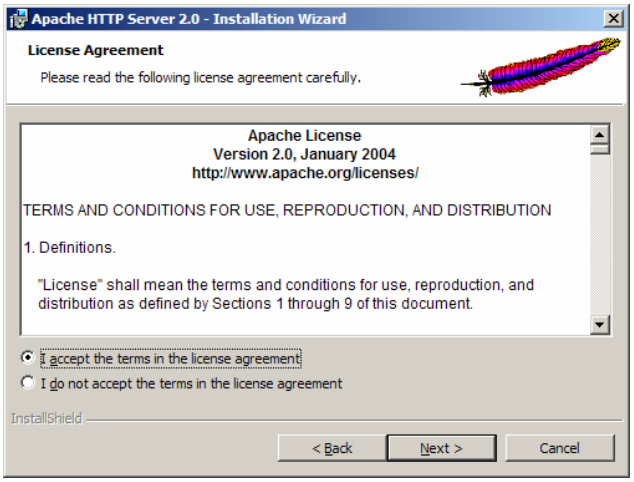


图 1-3　协议声明界面

在图 1-3 中，选中“I accept the terms in the license agreement”单选按钮，然后单击“Next”按钮，进入到下一个安装界面，如图 1-4 所示。

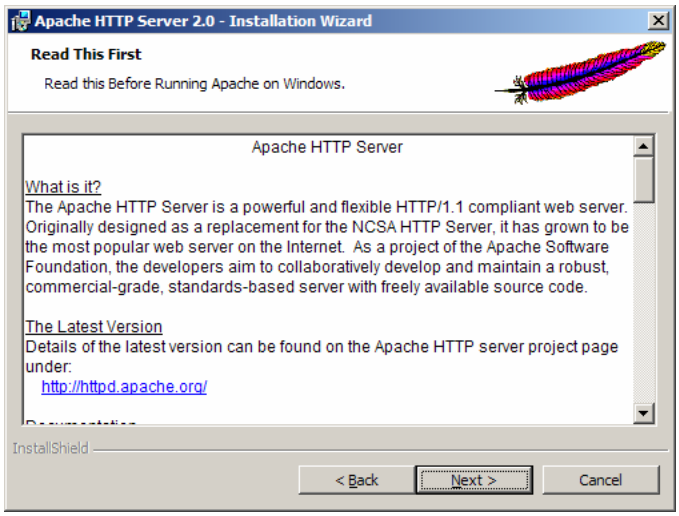


图 1-4　Apache 服务器说明

在图 1-4 中，只需要单击“Next”按钮，进入到下一个安装界面，如图 1-5 所示。

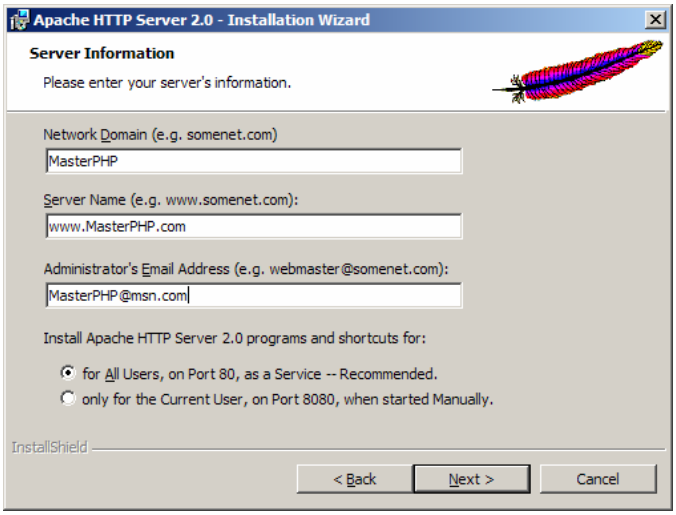


图 1-5　输入服务器信息

在图 1-5 中，需要输入服务器的相关信息，如网络域名、服务器名称以及管理员 E-mail 等，读者可以根据自己的情况输入。在下面的单选按钮中，选择第一项，表示对任何用户开放，同时设置服务器的侦听端口为“80”。如果选择了第二项，则只有本地用户可以连接和使用了。

提示　如果选择了第一项，用户同时安装了 IIS，则必须修改 IIS 的默认端口。否则会导致 Apache 无法正常工作，因为 IIS 的默认侦听端口也是 80。可以在 IIS 管理器中进行设置，或者也可以停止 IIS 服务器。

设置完成后，在图 1-5 中单击“Next”按钮，进入如图 1-6 所示的安装界面。

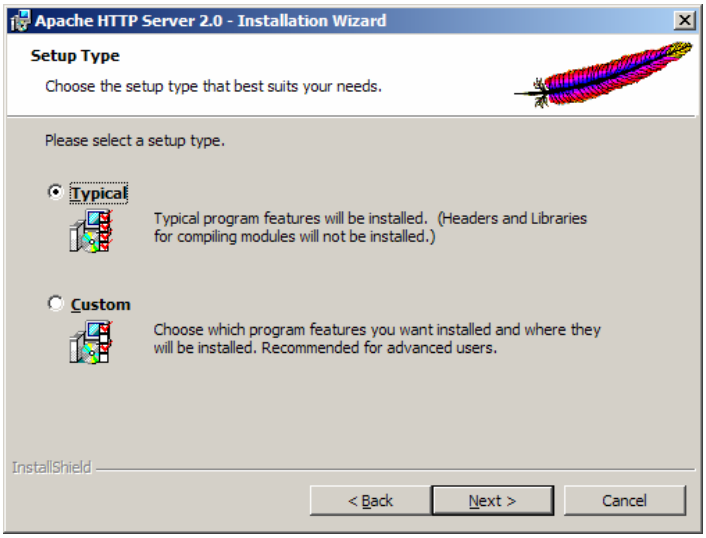


图 1-6　选择安装方式

在图 1-6 中可以选择安装方式，如“Typical”（典型）安装或者“Custom”（自定义）安装方式。对于初级用户，建议选择“Typical”（典型）安装方式。此处选择“Typical”方式，然后单击“Next”按钮，进入到下一个安装界面，如图 1-7 所示。

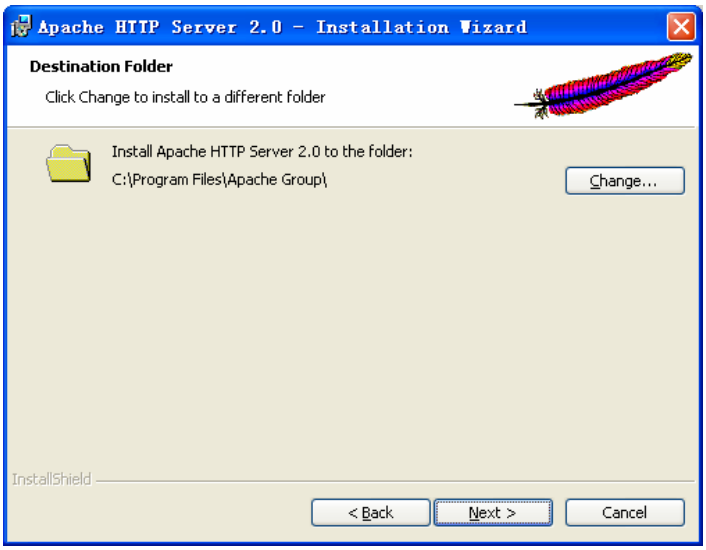


图 1-7　设置安装路径

在图 1-7 中，单击“Change…”按钮，可以修改安装路径。这里采用默认的安装路径“C:\Program Files\Apache Group”，然后单击“Next”按钮，进入下一个安装界面，如图 1-8 所示。

在图 1-8 中，单击“Install”按钮，进入下一个安装界面，如图 1-9 所示。

图 1-9 表示正在复制文件，文件复制完毕则安装完成，安装完成界面如图 1-10 所示。

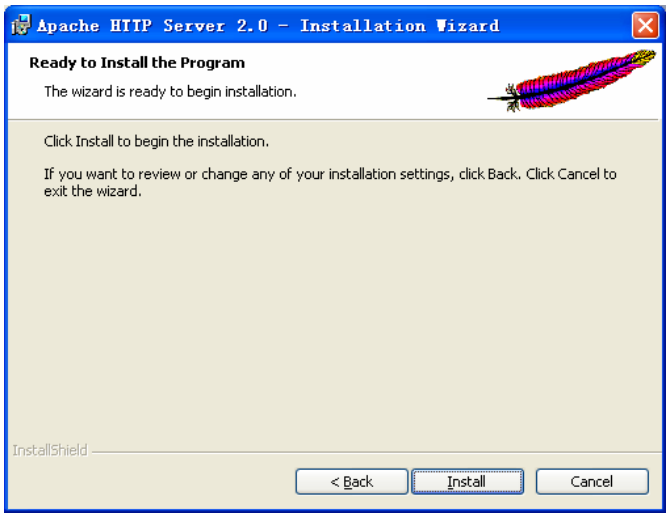


图 1-8　准备安装

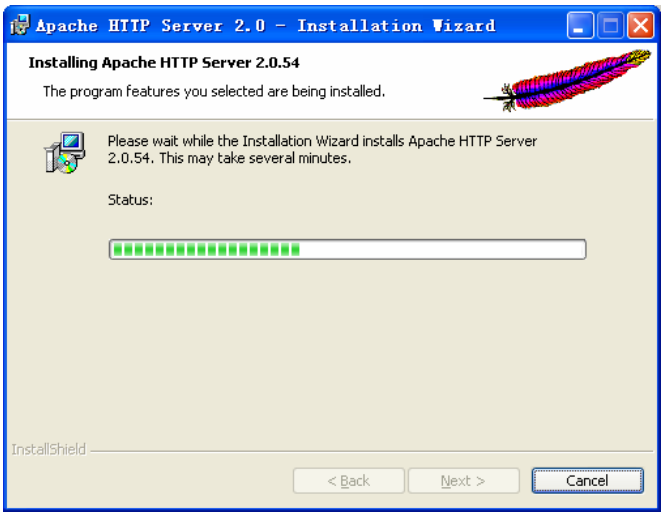


图 1-9　复制文件

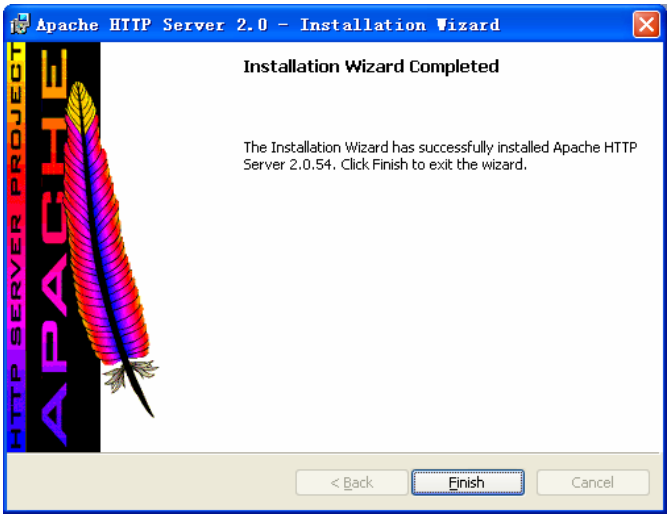


图 1-10　Apache 服务器安装完毕

服务器安装完毕会自动运行，运行后，显示在系统托盘中。如果用鼠标左键单击会弹出菜单，如图 1-11 所示。



图 1-11　Apache 左键菜单

在图 1-11 中，最左边的表示的是 Apache 服务器。其中单击菜单【Stop】表示停止 Apache 服务器，单击【Start】表示启动 Apache 服务器，而单击【Restart】表示重新启动服务器。

如果双击该图标，便会弹出 Apache 的服务监视器，如图 1-12 所示。

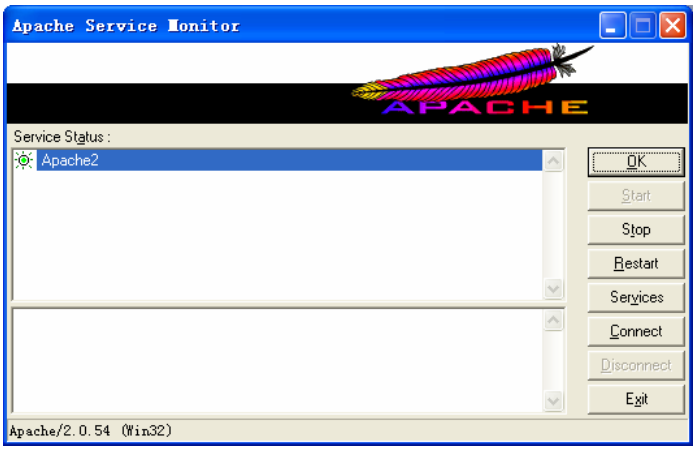


图 1-12　Apache 服务器监视器

在图 1-12 中，也可以进行启动和停止服务器等各种管理操作。通过以上步骤，Apache 服务器已经安装完成，下面通过浏览器来测试该服务器。

启动 IE，在浏览器中输入“http://localhost/”或者输入“http://127.0.0.1/”，如果可以看到如图 1-13 所示的界面，则表示安装成功。

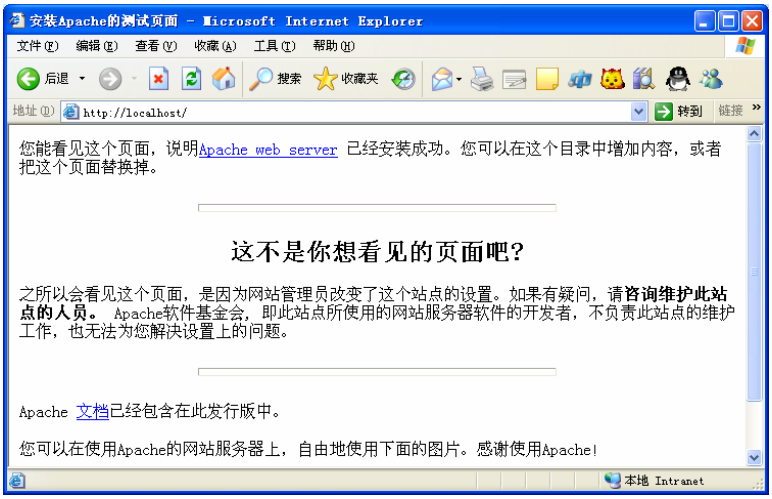


图 1-13　测试 Apache 服务器

接下来还需要简单配置 Apache 服务器，以便使用时可以方便地定位 PHP 文件。

通常默认的访问文件路径为“C:\Program Files\Apache Group\Apache2\htdocs”，用户只需要将自己的 HTML 文件拷贝到该目录下，即可访问该文件。首先来看一个简单的 HTML 文件，文件内容如代码 1-1 所示。

【代码 1-1】 HelloWorld.htm

```
<html>
<head>
<title>helloworld</title>
</head>
<body>
Hello World!
</body>
</html>
```

将上面的文本保存成“HelloWorld.htm”并拷贝到“C:\Program Files\Apache Group\Apache2\htdocs”目录下，然后在 IE 浏览器中输入“http://localhost/HelloWorld.htm”就可以看到输出结果，如图 1-14 所示。



图 1-14 HelloWorld.htm 执行结果

很多时候，为了方便文件的管理，需要将所有被访问的文件放在其他的文件夹中，比如需要将文件放在“d:\masterphp”目录中，然后通过 Apache 来访问，则必须要修改 Apache 的设置。

打开“C:\Program Files\Apache Group\Apache2\conf”目录下的 httpd.conf 文件，然后查找到“DocumentRoot”一行，如图 1-15 所示。

```
#
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
DocumentRoot "C:/Program Files/Apache Group/Apache2/htdocs"
```

图 1-15 修改后的 DocumentRoot 设置



提示 用记事本即可打开 httpd.conf 文件，该文件为文本文件。

将“DocumentRoot”的路径修改成需要设置的访问路径，这里设置成“d:/masterphp”，如图 1-16 所示。

```
#
DocumentRoot "d:/masterphp"
```

图 1-16 修改后的 DocumentRoot 设置



提示 这里使用的分隔符是斜杠“/”而不是“\”。

如果将 HelloWorld.htm 文件拷贝到“d:/masterphp/chap01”中，将文件名改成 Hello-World.htm，然后在 IE 中输入“http://localhost/chap01/Hello-World.htm”，仍将出现如图 1-14 所示的界面，则表示目前 Web 服务器根目录已经被修改到“d:/masterphp”了。

提示　Apache 服务器功能非常强大，能够进行各种配置。如果需要单独配置的，会在介绍相应内容时说明。同时关于 Apache 服务器的常用设置选项，会在本书的附录中提供。

1.4.2　PHP 的安装配置

在成功安装 Apache 服务器后，接下来需要安装 PHP。Apache 虽然能够支持多种的编程语言，但是需要单独安装对应的引擎，如果要让 Apache 支持 PHP，则必须安装 PHP 引擎。

同 Apache 服务器一样，PHP 也是免费软件，可以在“http://www.php.net/downloads.php#v5.1”中下载 PHP 5.0，到目前为止比较成熟的版本为 5.0.5，用户可以下载 5.1.0 测试版。下载界面如图 1-17 所示。

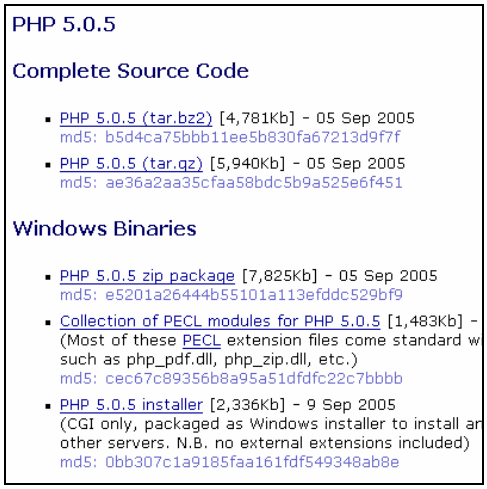


图 1-17　PHP 下载界面

可以下载“Windows Binaries”下面的“PHP 5.0.5 zip package”，将该文件保存到硬盘中。

下载后的文件是一个 ZIP 压缩文件，可以直接将其解压缩到一个目录下，不需要安装。这里将其解压缩到“d:\php5”目录下，如图 1-18 所示。

接下来要将 PHP 和 Apache 服务器连接上。具体操作步骤如下。

（1）修改“d:\php5”目录下的“php.ini-recommended”文件名，改成“php.ini”。

（2）将“php.ini”文件拷贝到 Apace 目录下，即“C:\Program Files\Apache Group\Apache2”下。

（3）打开 php.ini 文件，修改以下几个重要的关键字。

查询到文件中的“extension_dir = "./"”，如图 1-19 所示。

将“extension_dir = "./"”修改成 PHP 目录里 extensions 的路径，即“D:\PHP5\ext”，如图 1-20 所示。

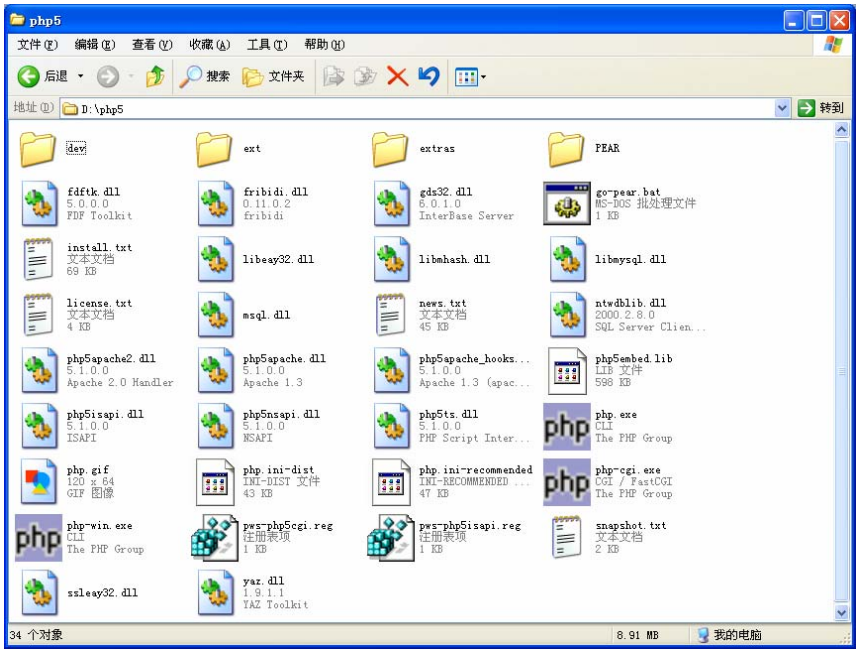


图 1-18　PHP 文件包

```
; Directory in which the loadable extensions (modules) reside.
extension_dir = "./"
```

图 1-19　修改前的 extension_dir 设置值

```
; Directory in which the loadable extensions (modules) reside.
 extension_dir = "D:\PHP5\ext"
```

图 1-20　修改后的 extension_dir 设置值

如果在显示中文的时候，出现乱码，可以修改 php.ini 中的参数 default_charset，在 php.ini 中找到“;default_charset = "iso-8859-1"”，改成“default_charset = "gbk"”，注意要去掉前面的“;”。

很多时候需要利用 session 对象来存储数据，如 session 身份验证、设计购物车等，因此还必须找到另外一个属性，即“session.save_path”。

原始设置如下：

```
;session.save_path = "/tmp"
```

将其修改为：

```
session.save_path = "d:/php5/session"
```



d:/php5/session 是需要手工创建的，读者也可以创建其他的目录，但必须去掉前面的分号。

查找 display_errors 选项和 cgi.force_redirect 选项。

原始设置如下：

```
display_errors = Off
; cgi.force_redirect = 1
```

修改后：

```
display_errors = ON
cgi.force_redirect = 0
```



display_errors 属性表示在运行时如果发生错误，是否会显示错误信息。而 cgi.force_redirect 为 0 表示强制不运行在 CGI 模式下。

PHP 在 Apache 中工作，有两种方式，一种是以 CGI 二进制文件方式工作，一种是使用 Apache 模块 DLL 的方式工作，采用不同的方式，需要在 php.ini 文件中进行不同的设置。

如果采用 CGI 方式，则需要将以下代码插入到 Apache 的 httpd.conf 文件中：

```
ScriptAlias /php/ "d:/php5/"
AddType application/x-httpd-php .php
Action application/x-httpd-php "/php/php.exe"
```



其中 d::/php5/是 PHP 的安装路径。

如果采用模块化的工作方式，则首先需要将“d:/php5/”下的 php5ts.dll 文件拷贝到系统的 winnt/system32（Windows NT/2000）或 windows/system32（Windows XP）；接着将“d:/php5/”下的 php5apache2.dll 文件复制到“C:\Program Files\Apache Group\Apache2\modules”下，然后将下面的一段代码插入到 httpd.conf 文件的最后。

```
LoadModule php5_module modules/php5apache2.dll
AddType application/x-httpd-php .php
```



如果还想让 Apache 执行其他后缀名，则可以添加其他的类型，如下所示。

AddType application/x-httpd-php .php4
AddType application/x-httpd-php .php3
AddType application/x-httpd-php .phtml
AddType application/x-httpd-php-source .phps

Apache 默认的连接文件名为“index.html”，即如果在 IE 中输入“http://localhost/”和输入“http://localhost/index.html”显示的结果是一致的，如果想增加其他的默认连接文件名，则在 httpd.conf 文件中找到“DirectoryIndex”关键字。

原始设置如下：

```
DirectoryIndex index.html index.html.var
```

如果想支持 default.php，则修改成如下的格式：

```
DirectoryIndex default.php index.html index.html.var
```



本书将以模块化的方式来执行 PHP，本书光盘中将会提供配置好的 php.ini 和 httpd.conf 文件。

为了测试 PHP 配置是否成功，打开记事本，输入以下代码：

【代码 1-2】　version.php

```
<?php
    phpinfo()
?>
```

将该段代码保存成 version.php（后缀名必须为 php），然后停止 Apache 服务器，再启动。将 version.php 拷贝到“d:\masterphp”目录下。然后在 IE 中输入“http://localhost/chap01/version.php”，

如果一切配置正常，显示的结果如图 1-21 所示。

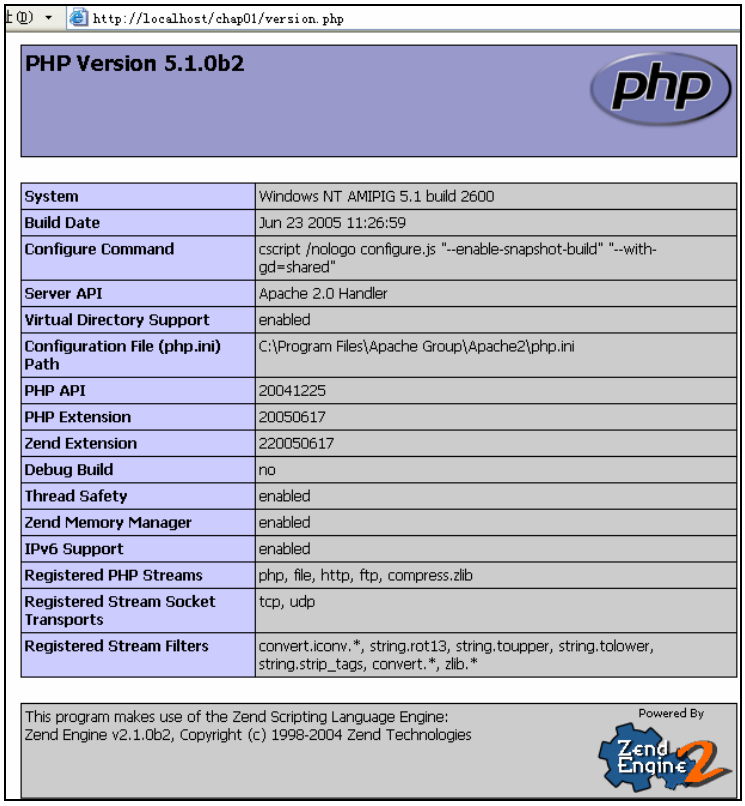


图 1-21　version.php 执行结果

提示　在该页面中，用户可以查看大多数的系统配置，如数据库配置、环境配置等。

至此，PHP 基本配置完成。但是 PHP 还有很多的配置选项，本书将在附录中给出常用的设置说明。

1.4.3　MySQL 的安装配置

本书将会介绍 PHP 同各种数据库的连接编程技术，但是重点介绍 MySQL，有关 SQL Server、Oracle 等大型数据库的安装和配置，读者可以参考其他书籍。

同 Apache、PHP 一样，MySQL 也是一个免费软件，目前 MySQL 的最新版本为 5.0。

读者可以在“www.mysql.com”网站下载 MySQL 软件，尽量下载可直接安装的版本。下载完成后，双击安装文件，进入安装向导，如图 1-22 所示。

在图 1-22 中，单击“Next”按钮，进入下一个安装界面，如图 1-23 所示。

在图 1-23 中，可以选择“Typical”（典型）安装、“Complete”（完全）安装、“Custom”（自定义）安装。建议初学者选择“Typical”（典型）安装或者“Complete”（完全）安装。本书选择“Complete”安装，然后单击“Next”按钮，进入到下一个安装界面，如图 1-24 所示。



图 1-22　MySQL 安装向导

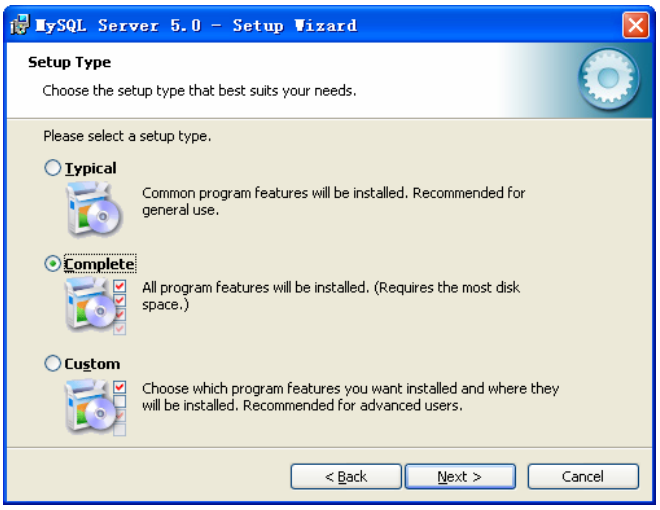


图 1-23　选择安装类型

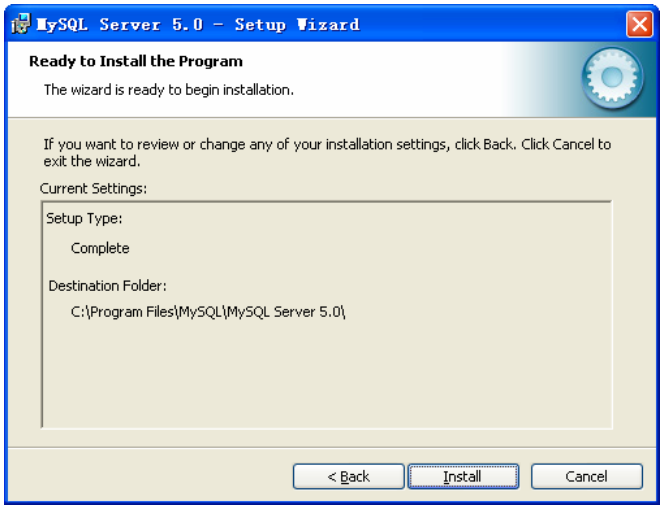


图 1-24　安装说明

在图 1-24 中，单击“Next”按钮，进入文件安装拷贝阶段，如图 1-25 所示。

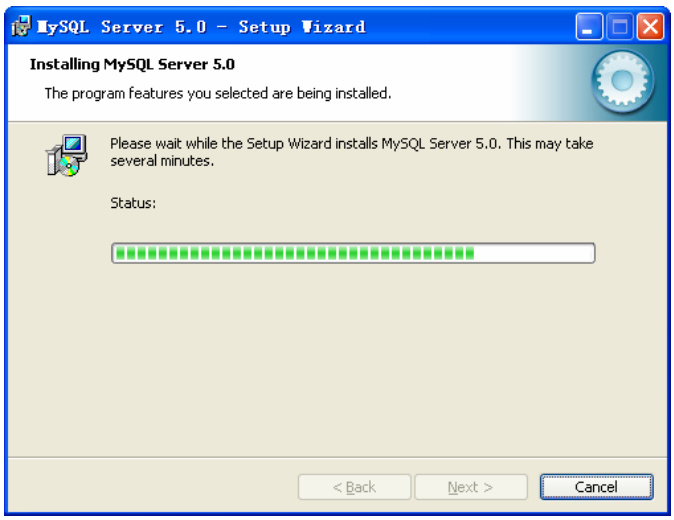


图 1-25　文件复制

提示　如果选择典型或者完全安装，将不能设置安装路径，默认的安装路径为“C:\Program Files\MySQL\”，如果要改变安装路径，可以选择自定义安装。

文件复制完成后，进入 MySQL 的账号设置界面，如图 1-26 所示。

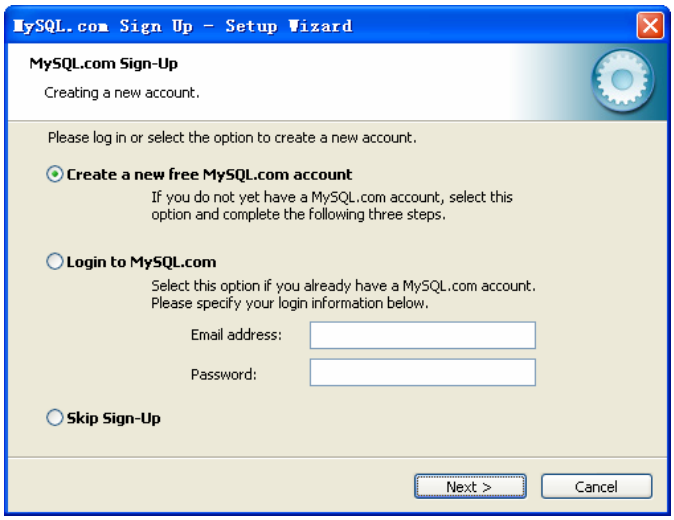


图 1-26　注册 MySQL 账号

在图 1-26 中，可以选择注册一个新的 MySQL 账号或者登录到已经注册的账号，也可直接跳过注册。此处选择第一项，创建一个新的账号，然后单击“Next”按钮，进入到下一个设置界面，如图 1-27 所示。

在图 1-27 中输入 E-mail 地址以及密码，然后单击“Next”按钮，进入到下一个安装界面，如图 1-28 所示。

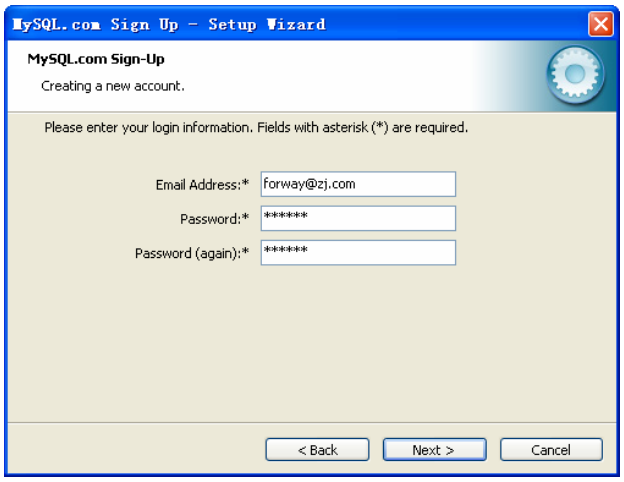


图 1-27　设置 E-mail 地址和密码

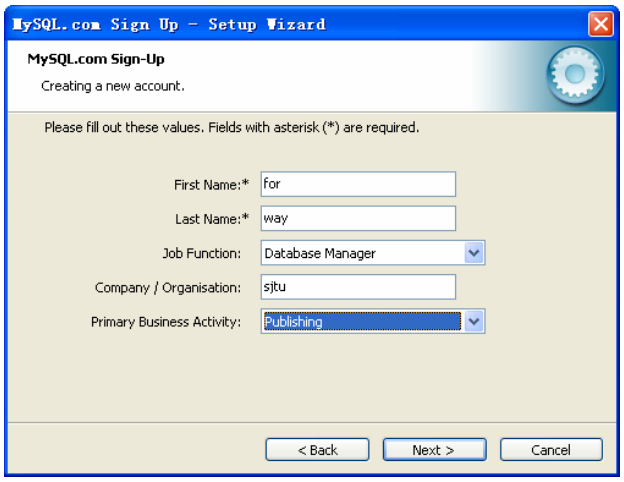


图 1-28　继续输入注册信息

在图 1-28 中，输入姓名和公司名称等信息，然后单击“Next”按钮，进入下一个设置界面，如图 1-29 所示。

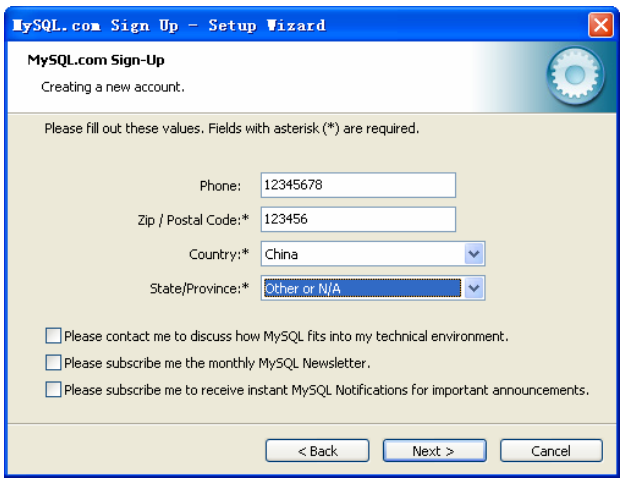


图 1-29　其他注册信息

在图 1-29 中，输入联系电话、邮政编码以及国籍等信息，然后单击“Next”按钮，进入到下一个设置界面，如图 1-30 所示。

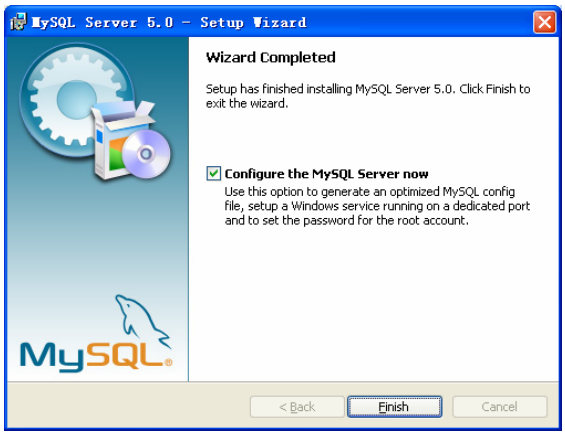


图 1-30　安装完成

在图 1-30 中，选中“Configure the MySQL Server now”复选框，然后单击“Finish”按钮，进入 MySQL 数据库服务器设置界面，如图 1-31 所示。

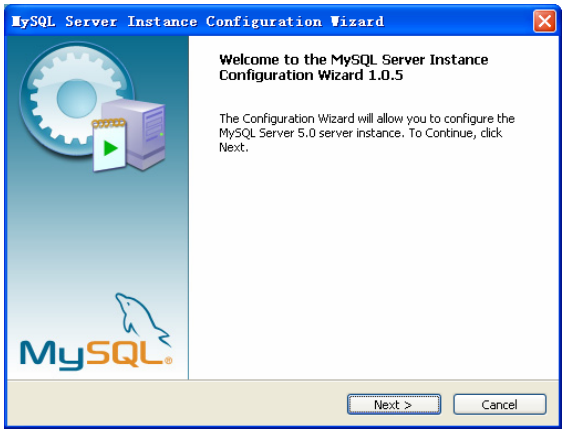


图 1-31　MySQL 服务器设置向导

在图 1-31 中，单击“Next”按钮，进入到下一个设置界面，如图 1-32 所示。

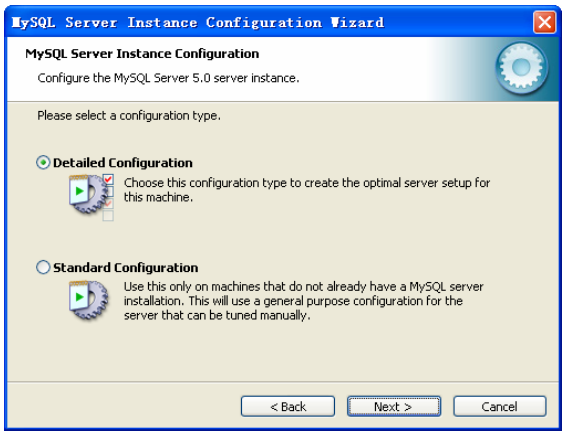


图 1-32　选择设置类型

在图 1-32 中，可以选择设置类型，第一个单选按钮是详细的设置；第二个单选按钮是标准设置。此处选择第一项，即详细设置，单击“Next”按钮，进入到下一个设置界面，如图 1-33 所示。

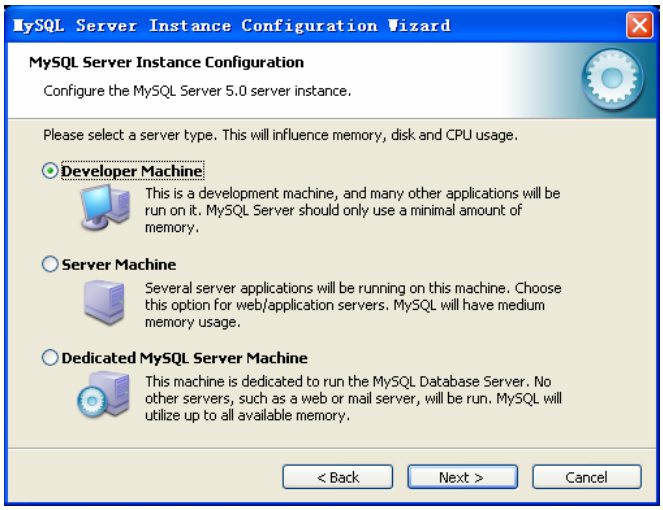


图 1-33　选择服务器类型

在图 1-33 中，可以选择服务器类型。其中第一个单选按钮表示运行 MySQL 的计算机，主要用于开发；第二个单选按钮表示主要用于后台服务器；而第三个单选按钮表示该计算机只运行 MySQL 服务器，而不运行其他服务器。这里选择第一项，即开发类型，然后单击“Next”按钮，进入到下一个设置界面，如图 1-34 所示。

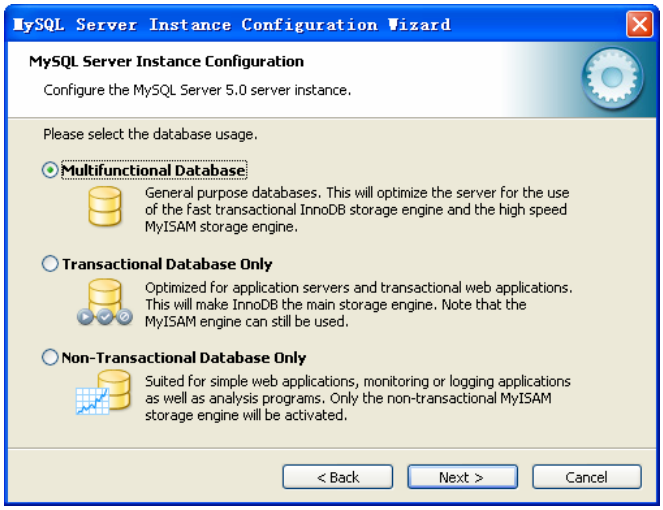


图 1-34　选择数据库用途

在图 1-34 中，可以设置服务器的用途，第一项表示该数据库是多功能数据库；第二项表示数据库只用于进行事务处理；第三项表示只用于简单的网络开发。这里选择第一项，然后单击“Next”按钮进入下一个设置界面，如图 1-35 所示。

在图 1-35 中，可以设置默认的数据库存放路径，这里设置的路径为“d:\MasterPHP\databae”。单击“Next”按钮，进入下一个设置界面，如图 1-36 所示。

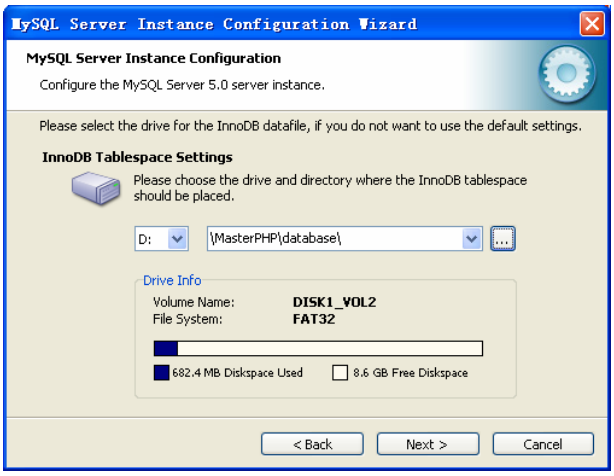


图 1-35　设置默认数据库文件存放路径

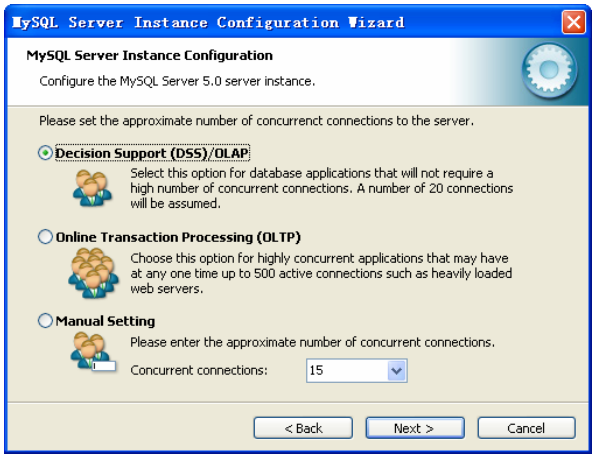


图 1-36　设置最大连接的客户数量

在图 1-36 中，可以设置最大连接客户数量，第一个选项表示一般为 20 个客户；第二个选项表示可以为 500 个客户；第三个选项表示为手动设置。这里选择第一个选项，然后单击“Next”按钮，进入下一个设置界面，如图 1-37 所示。

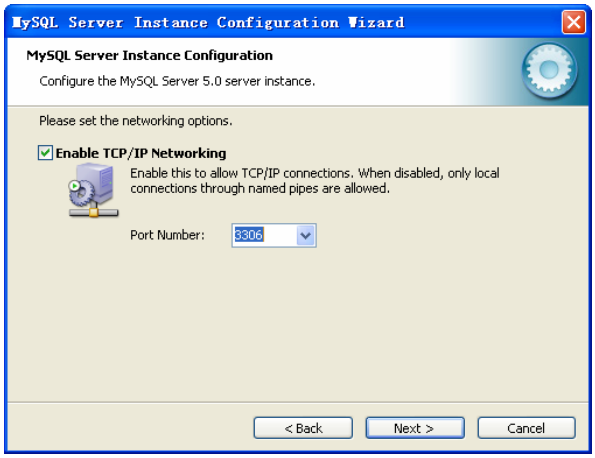


图 1-37　MySQL 网络设置

在图 1-37 中，可以设置是否运行网络连接，如果不允许网络连接，则本地机器通过命名管道才能访问，因此需要选中“Enable TCP/IP Networking”复选框，然后单击“Next”按钮进入下一个设置界面，如图 1-38 所示。

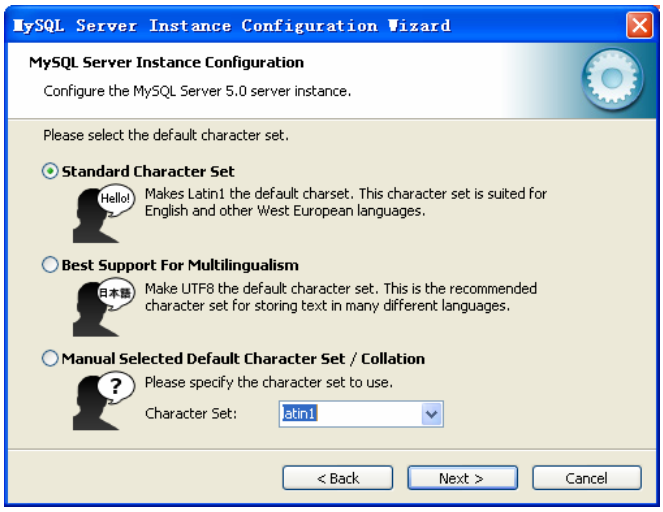


图 1-38　设置默认字符集

在图 1-38 中，可以设置默认字符集，第一项表示将“Latin1”设置为默认字符集；第二项表示将“UTF8”设置为默认字符集；第三项表示手动设置字符集，这里设置为第一项，然后单击“Next”按钮，进入下一个设置界面，如图 1-39 所示。

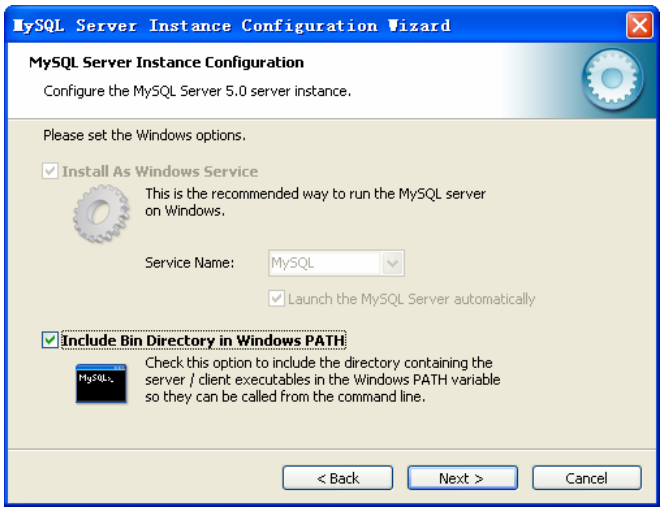


图 1-39　设置 Windows 选项

在图 1-39 中，可以设置 Windows 选项。一般需要选中第一个复选框，表示将 MySQL 作为 Windows 服务；如果选中第二项，则可以通过命令行来启动 MySQL 数据库。这里同时选中两个复选框，然后单击“Next”按钮，进入下一个设置界面，如图 1-40 所示。

在图 1-40 中，可以设置 MySQL 的运行密码，选中第一个复选框，然后输入密码；如果选中第二个选项，表示创建一个匿名账号，这样会让数据库系统不安全。单击“Next”按钮，进入下一个设置界面，如图 1-41 所示。

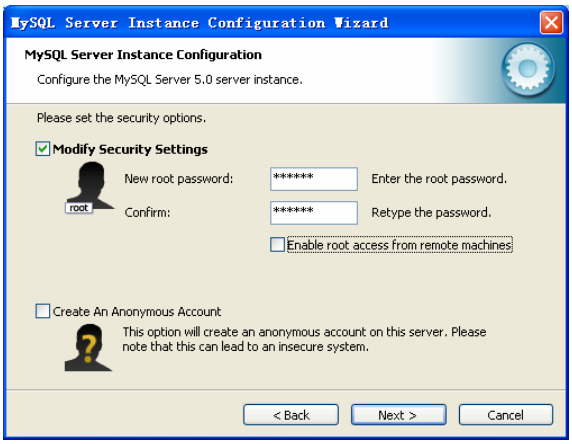


图 1-40　安全选项

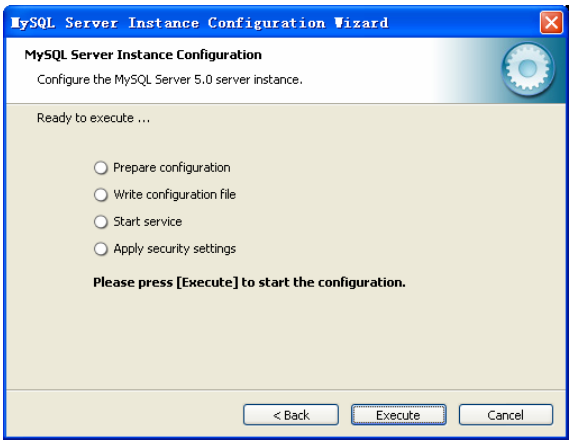


图 1-41　准备进行设置



这里设置的密码为 123456，在以后的编程中，都需要使用该密码，为了配合本书中源代码的使用，建议读者也使用该密码。

在图 1-41 中，单击“Execute”按钮，可以开始各种具体的设置。设置完成后的界面如图 1-42 所示。

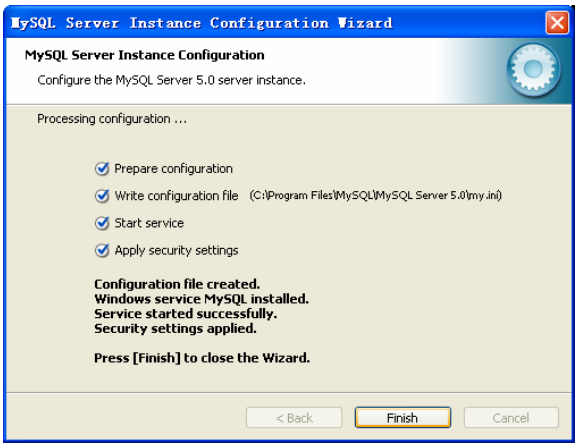
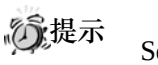


图 1-42　MySQL 设置完成



如果需要修改 MySQL 设置，可以选择【开始→所有程序→MySQL→MySQL Server 5.0→MySQL Server Instance Config Wizard】，重新设置。

经过以上步骤，MySQL 已经设置完毕。接下来需要测试 MySQL 数据库是否安装成功。

可以选择【开始→所有程序→MySQL→MySQL Server 5.0→MySQL Command Line Client】进入到 MySQL 的客户端，如图 1-43 所示。

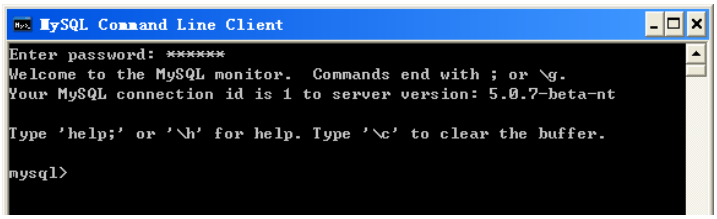


图 1-43　MySQL 客户端

要进入到该客户端，首先要输入根用户“root”的密码，密码为图 1-40 中设置的密码。如果输入密码后出现了“mysql>”的标志，则表示数据库安装成功。



因为在图 1-39 中已经设置了可以通过命令行来执行 MySQL，因此通过菜单【开始→运行】，在弹出的对话框中输入“cmd”命令，然后在命令模式下输入“mysql –h localhost –u root –p”命令，也能够进入到 MySQL 的客户端，其中 localhost 为服务器名称，root 为用户名。

MySQL 安装成功后还需要同 PHP 连接上，因此，还需要进一步的配置。

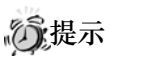
打开“C:\Program Files\Apache Group\Apache2”目录下的 php.ini 文件，查找“MySQL”关键字，修改以下几个属性。

原始设置为：

```
mysql.default_port =
mysql.default_port =
mysql.default_user =
mysql.default_password =
```

修改为：

```
mysql.default_port = 3306
mysql.default_host = localhost
mysql.default_user = root
mysql.default_password = 123456
```



default_port 表示 MySQL 的网络端口，安装的时候设置为 3306。

default_host 为服务器名称，localhost 表示本地机器。

default_user 表示用户名，root 是默认的用户名。

default_password 为安装时设置的密码，这里为 123456。

修改了以上属性之后，还需要启动 PHP 对 MySQL 的支持模块，在 php.ini 文件中查找“extension=php_mysql.dll”字符串。

原始设置为：

```
;extension=php_mysql.dll
```

将其修改为：

```
extension=php_mysql.dll
```

即去掉前面的分号即可。



PHP 5.0 中增加了 mysqli 函数库对 MySQL 的支持，读者也可以使用该库。具体设置请参照 MySQL 编程的相关内容。

下面通过简单的代码测试 MySQL 是否安装成功。测试代码如代码 1-3 所示。

【代码 1-3】 test-mysql.php

```
<?php
$link=mysql_connect('localhost','root','123456');
if(!$link) echo "失败!";
else echo "成功!";
mysql_close();
?>
```

在 IE 浏览器中输入“http://localhost/test-mysql.php”，如果连接成功，则会在浏览器中显示“成功!”，如图 1-44 所示。

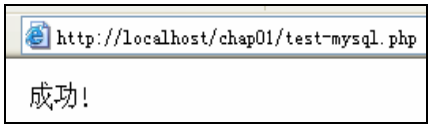


图 1-44 测试 PHP 连接 MySQL

经过以上各种配置之后，本书所涉及的基本的调试和学习环境已经搭建成功，接下来会带领读者学习 PHP 的各种编程知识。



配置 Apache+PHP+MySQL 的方法有好多种，但是本质上都差不多，此处只介绍了其中的一种方法，如果读者有兴趣，可以参考网络上的其他配置方法。

第 2 章　PHP 语言轻松入门

本章主要介绍 PHP 网络开发的基础知识，包括 PHP 语言基础、PHP 变量、PHP 运算符、PHP 控制语句、PHP 数组和 PHP 的函数等。

2.1　PHP 语言基础

熟悉 ASP 读者一定知道，ASP 代码是嵌入在 HTML 代码中的，而 PHP 代码也一样，是嵌入在 HTML 代码中的，然后通过一定的标记来区分 HTML 代码和客户端脚本。

2.1.1　PHP 文件格式

PHP 文件格式非常简单，本质上而言就是一个文本文件。因此用户可以通过任何文本编辑工具，如记事本、Dreamweaver 等工具来输入 PHP 代码，然后将其保存成后缀为“.php”文件。

PHP 文件事先不需要经过编译，只要按照 1.4.2 小节介绍的方法进行 PHP 环境配置，然后将文件放置在事先设定好的目录中，就可以在网络浏览器中浏览文件了。

一个完整的 PHP 文件由以下元素构成：

（1）HTML 代码；

（2）PHP 标记（分隔符）；

（3）PHP 代码；

（4）注释；

（5）空格等。

下面来学习本书的第一个完整的程序。

实例 2-1　Hello World 程序

实例 2-1 主要实现的功能是在浏览器中输出“Hello,World”的语句，如代码 2-1 所示。

【代码 2-1】　helloworld.php

```
<html>
<head>
<title>Hello World!</title>
</head>
<body>
<?php
    //输出 Hello,World
    echo "Hello,World";
?>
```

```
</body>
</html>
```

代码 2-1 中，“<html>”等表示是 HTML 代码、“<?php…?>”表示是 PHP 标记、“echo "Hello,World";”表示的是 PHP 代码、“//输出 Hello,World”表示的是代码的注释。



本书第 1 章也介绍了 Hello World 程序，但是文件的后缀名是.htm，其中不含有 PHP 代码。而实例 2-1 的 HelloWorld 是由 PHP 代码输出的。

2.1.2 PHP 标记

前面提到了，PHP 代码是嵌入在 HTML 文件中的。HTML 代码中有普通文本，还有许多 HTML 的标记元素，如“hello”，表示在浏览器中输入加粗的“hello”字符串，而“”被浏览器解释后并不输入，这里的“”就是 HTML 的标记元素，HTML 标记元素是通过“<”和“>”来区分于普通文本的。

由于 PHP 嵌入在 HTML 中，因此也需要标记对（分隔符）来区分。通常情况下，有以下几种方式：

（1）<?php . . . ?>

（2）<? . . . ?>

（3）<script language="php"> . . . </script>

（4）<% . . . %>



当使用“<? . . . ?>”将 PHP 代码嵌入于 HTML 文件中时，可能会同 XML 发生冲突。能否使用这一缩写形式还取决于 PHP 本身的设置。为了适应 XML 和其他编辑器，可以在开始的问号后面加上“php”使 PHP 代码适应于 XML 分析器，如“<?php. . . ?>”。也可以像写其他脚本语言那样使用脚本标记，如“<script language="php"> . . . </script>”。本书推荐使用<?php…?>的模式来进行 PHP 编码。

由于 ASP 的分隔符为“<%...%>”，所以为了区别 ASP 和 PHP，尽量少使用该分隔符。

下面简单举例说明 PHP 标记的用法。

实例 2-2 <?php…?>的应用

实例 2-2 演示了如何使用<?php…?>标记输出内容，如代码 2-2 所示。

【代码 2-2】 php1.php

```
<?php echo("第一种书写方法!\n"); ?>
```

输出结果如下：

```
第一种书写方法!
```

实例 2-3 <script language="php"></script>的应用

实例 2-3 演示了如何使用“<script language="php"> ... </script>”标记输出内容，如代码 2-3 所示。

【代码 2-3】　php2.php

```
<script language="php">
    echo ("script 书写方法!");
</script>
```

输出结果如下：

```
script 书写方法!
```

实例 2-4　<%...%>的应用

“<%...%>”是 ASP 语言的标记。如果需要在 PHP 中使用这种标记，需要修改 php.ini 文件中的设置。

将 php.ini 文件中的如下文本：

```
; Allow ASP-style <% %> tags.
asp_tags = Off
```

改成：

```
; Allow ASP-style <% %> tags.
asp_tags = ON
```

将 asp_tags 的值由 OFF 改成 ON。然后重新启动 Apache 服务器。

<%...%>的标记应用如代码 2-4 所示。

【代码 2-4】　php3.php

```
<%
    echo("这是 ASP 的标记输出");
%>
```

输出结果如下：

```
这是 ASP 的标记输出
```

实例 2-5　<?...?>的应用

“<?...?>”标记是“<?php…?>”标记的简写。但是如果要在代码中使用 XML，则最好不要使用这种格式。如果要使用这种简写标记，需要将 php.ini 文件中的“short_open_tag=OFF”修改成“short_open_tag = ON”。然后重新启动 Apache 服务器。

<?...?>标记的应用如代码 2-5 所示。

【代码 2-5】　php4.php

```
<?
    echo("这是 PHP 的简写标记输出");
?>
```

输出结果如下：

```
这是 PHP 的简写标记输出
```

实例 2-6　PHP 控制 HTML 输出

本实例演示了如何使用 PHP 代码控制 HTML 代码的输出，如代码 2-6 所示。

【代码 2-6】　php5.php

```
<?php
```

```
$expression=1;
if ($expression) {
   ?>
   <strong>这是真的!</strong>
   <?php
} else {
   ?>
   <strong>这是假的!</strong>
   <?php
}
?>
```

输出结果如下：

```
这是真的!
```

提示 以上代码中使用了变量、条件判断语句以及 echo 函数，在后面会详细介绍。

2.1.3 PHP 文件的访问形式

在学习 PHP 语言之前，首先要明确一个概念，PHP 语言是一种服务器端执行的语言，如果要通过浏览器访问网页，必须先经过服务器的执行，才能看到正确的结果，这同普通的 HTML 文件是不一样的，因此访问 PHP 文件，必须通过虚拟路径来访问。而普通的 HTML 文件，既可以通过物理路径访问，也可以通过虚拟路径访问。

下面通过具体实例来说明。

实例 2-7 HTML 文件的访问方法

代码 2-7 是一个简单的 HTML 格式的文件，文件名为 simplehtml.htm。

【代码 2-7】 simplehtml.htm

```
<Html>
<head>
<title>
Hello World!
</title>
</head>

<body>

<b>hello world!</b>

</body>
</html>
```

将该文件保存成 simplehtml.htm，保存路径为“d:\MasterPHP\chap02\simplehtml.htm”。其中“d:\MasterPHP”是笔者设置的 Apapche 服务器的根目录。

（1）直接使用“d:\MasterPHP\chap02\simplehtml.htm”物理路径访问该文件，结果如图 2-1 所示。

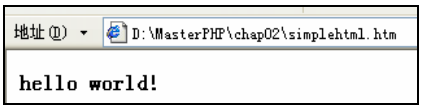


图 2-1　使用物理路径访问 HTML 文件

（2）使用“http://localhost/chap02/simplehtml.htm”虚拟路径来访问该文件，结果如图 2-2 所示。



图 2-2　使用虚拟路径访问 HTML 文件

提示　注意图 2-1 和图 2-2 中地址栏中的访问路径。

实例 2-8　PHP 文件的访问方法

本实例通过虚拟路径来访问 PHP 文件，实例文件名为 helloworld.php，即代码 2-1 所示的文件。要运行该程序，则必须在浏览器地址栏中输入“http://localhost/chap02/ helloworld.php”才能得到正确的结果。helloworld.php 文件的执行结果如图 2-3 所示。



图 2-3　访问 PHP 文件

提示　之所以通过虚拟路径访问 PHP 文件，是因为 PHP 文件代码必须经过服务器解释后才能被发送到客户端。而 PHP 代码是不能被客户端查看到的。

为了说明 PHP 程序是被服务器解释执行后发送到客户端，可以在图 2-3 中右键单击浏览器，然后在弹出菜单中选择【查看源文件】，可以看到源文件代码如下所示：

```
<Html>
<head>
<title>Hello World!</title>
</head>
<body>
Hello,World</body>
</html>
```

从上面代码可以看出，PHP 代码实际上并没有被发送到客户端，而是被服务器执行后以普通 HTML 的格式发送到客户端了。

2.1.4　PHP 程序注释

任何一种编程语言，都少不了对代码的注释。因为一个比较好的应用程序源代码，代码注释总是非常详细的。良好的代码注释对后期维护、升级能够起到非常重要的作用。

在 PHP 的程序中，加入注释的方法很灵活。可以使用 C 语言、C++语言或者是 UNIX 的 Shell 语言的注释方式，还可以混合使用。

下面通过实例来学习 PHP 程序注释。

实例 2-9　PHP 程序注释

本实例使用了多种方法来对 PHP 代码进行注释，如代码 2-8 所示。

【代码 2-8】　rem.php

```
<?php
    echo "C++语言注释的方法即  // <br>"; // 采用 C++的注释方法
    /* 多行注释
    对于大段的注释很有用的哦 */
    echo "多行注释方法即  /*...*/ <br>";
    echo "Unix 的注释方法即  # <br>"; # 使用 UNIX Shell 语法注释
?>
```

代码的执行结果如图 2-4 所示。

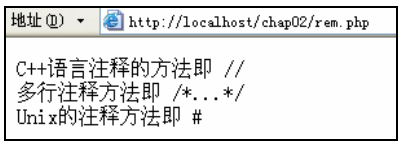


图 2-4　PHP 程序注释

从代码可以看出，在 PHP 中可以通过“//”或者“#”来进行单行代码的注释，同时还可以通过“/*…*/”进行大段代码的注释。

提示　在进行多行注释的时候，不能嵌套使用“/*…*/”注释符号，如代码 2-9 的注释方法是错误的。

【代码 2-9】　err_rem.php

```
<?php
/*
echo "不能嵌套使用多行注释符号\n"; /* 嵌套使用会出错 */
*/
?>
```

如果执行该段代码，则会出现如下错误提示：

```
Parse error: parse error, unexpected '*' in D:\MasterPHP\chap02\err_rem.php
on line 4
```

2.1.5　PHP 语法概述

PHP 语法主要借鉴于 C/C++，也部分参考了 Java 和 Perl，因此熟悉 C/C++的读者，可以很快地掌握 PHP 的语法。

类似 C 语言，在书写 PHP 代码的时候，每写完一句完整代码后，都要以分号“;”结束。代码 2-2 中的“echo "Hello,World";”语句就是一个完整的 PHP 语句，最后以“;”结束。而对于控制语句，一般不需要以分号“;”结束，如下代码：

```
if (a==b)
    echo "你们一样大";
```

其中“if (a==b)”语句后面不需要跟分号。如果控制语句下面有多行代码，则必须使用大括号“{…}”扩起来，如下所示：

```
if (a==b)
{
    echo "你们一样大";
    echo "希望你们考上好学校";
}
```


提示　PHP 代码书写的基本知识，将会在后面通过实例来介绍。

既然是一门编程语言，肯定会由多种编程元素组成，如变量、常量、运算符、控制语句、数组、字符串、函数和对象等。下面将会详细介绍 PHP 语言中各种语法的概念及使用。

2.2　PHP 的变量

任何一种编程语言都有变量的定义。抽象地说，变量就是一个保存了一小块数据的“对象”。从变量的字面意思理解，表示该数据块中的值是随时都可以改变的，即在不同时间段内代表不同的实体。

2.2.1　PHP 的变量命名

一般，每种编程语言都会遵循变量声明的某些规则。这些规则包括变量的最大长度、是否能够包含数字或者字母字符、变量名是否能够包含特殊字符以及变量名是否能够以数字开头。

在 PHP 中，对变量名的长度没有任何限制，在变量名中可以使用数字和字母字符，但是同时也有以下约定：

（1）PHP 的变量名是区分大小写的，这和 C 语言是一致的；

（2）变量名必须是以美元符号（$）开始；

（3）变量名开头可以以下划线开始；

（4）变量名不能以数字字符开头。

下面的变量命名是合法的：

```
$myName
$_Age
```


提示　在给变量命名的时候，最好让变量具有一定的含义，能够代表一定的信息，这样有利于阅读源代码，同时也有利于对变量名的引用。

2.2.2　PHP 的数据类型

通常每个变量总会代表着一种数据类型。PHP 是一种类型化较弱的语言，这意味着变量可以含有任意给定的数据类型，该类型取决于使用变量的上下文环境。

有过编程经验的读者知道，在 C 语言或者 Java 语言中，声明一个变量时，该变量就具有

某一种类型，而且只能给该变量赋予这种类型的数据。如下：

```
Int tempNum=100;  //这种赋值是正确的
```

而如果要写成：

```
Int tempNum="hello,world" ; //这样编译器便会提示错误
```

在上面的两个语句中使用了“int”关键字来设定变量“tempNum”为整型数据。而在 PHP 中，对变量的数据类型不需要声明，可以直接为其赋值，如下所示：

```
$tempNum=100;
```

事实上，PHP 中变量数据类型的定义是通过为变量赋值（初始化），由系统自动设定的，如：

```
$tempNum=100;  //表示$tempNum 为整型
$tempStr="Hello,World"  //表示$tempStr 为字符串型
```

虽然在 PHP 中不用显式的为变量设定数据类型，但是还是设定了多种数据类型。PHP 中的数据类型分成两种，一种是标量数据类型，这是编程语言种最常见的简单数据类型；另一种表示复合数据类型，即将多个简单数据类型组合在一起，并将它们存储在一个变量名中。在 PHP 中，标量数据类型有如下几种：

（1）布尔型（boolean）；

（2）整型（integer）；

（3）浮点型（float）（浮点数，也做“double”）；

（4）字符串（string）。

复合数据类型有如下几种：

（1）数组（array）；

（2）对象（object）。

另外，PHP 中，还有两种特殊的数据类型：

（1）资源（resource）；

（2）空值（NULL）。

下面分别介绍这些数据类型。

1．布尔型

在所有的 PHP 变量中，布尔变量是最简单的变量。布尔变量保存一个 True 或者 False 值。其中 True 或者 False 是 PHP 的内部关键字。

要设定一个变量为布尔型时，只需要将 True 或者 Fasle 赋值给该变量，如下所示：

```
$AreYouHappy=True;
```



虽然 PHP 中自定义变量的名称是区分大小写的，但是包括 True 和 False 在内的 PHP 关键字和函数名是不区分大小写的。

当需要用一个变量来测试逻辑或者保存 True 和 False 值时，布尔变量就是解决方法。



在 PHP3 以及以前版本中，布尔数据类型是不可用的。

需要强调的一点就是，在 PHP 中，即使是 PHP 的关键字，也可以作为变量名，而在 C++ 等语言中，是不允许的。

实例 2-10　PHP 关键字作为变量

本实例演示了在 PHP 中如何将关键字 True 作为变量进行编程，如代码 2-10 所示。

【代码 2-10】　key.php

```
<?php
    $True="我是变量 True";
    echo($True);
    echo("<BR>");
    echo("我是关键字" . True)
?>
```

代码 2-10 的执行结果如图 2-5 所示。



图 2-5　关键字作为变量

提示　True 和 False 实际上代表数值 1 和 0，因此 True 在输出时显示为 1。但是在给变量赋值的时候，如果赋值为 1，则被当成整型，如果被赋值为 True，则被当成布尔型。

当转换为布尔型时，以下值被认为是 False：

（1）布尔值 False；

（2）整型值 0（零）；

（3）浮点型值 0.0（零）；

（4）空白字符串和字符串“0”；

（5）没有成员变量的数组；

（6）没有单元的对象（仅适用于 PHP 4）；

（7）特殊类型 NULL。

而所有其他值都被认为是 TRUE（包括任何资源）。通常布尔值可以用一些表达式来返回。如“a>b”、“a=b”等，可以在条件语句中应用。

提示　–1 和其他非零值（无论正负）一样，被认为是 True。

2．整型

整数数据类型只能包含整数。这些数据类型可以是正数也可以是负数。可以在数字前面加上“–”符号来表示负整数。在 32 位的操作系统中，它的有效范围是–2 147 483 648 到+2 147 483 647。在给一个整型变量赋值的时候，可以采用十进制、十六进制或者八进制形式来指定。

实例 2-11　为整型变量赋值

本实例通过 3 种方法为整型变量赋值，如代码 2-11 所示。

【代码 2-11】　integer.php

```
<?php
    $int_D=2147483647;  //十进制赋值
    echo($int_D);
```

```
    echo("<br>");
    $int_H=0x7FFFFFFF;  //十六进制赋值
    echo($int_H);
    echo("<br>");
    $int_O=017777777777;  //八进制赋值
    echo($int_O);
    echo("<BR>");
?>
```

代码执行结果如图 2-6 所示。



图 2-6　为整型变量赋值

从图 2-6 中可以看出，3 种方式进行整型数据赋值的效果是一样的。

提示　如果指定一个数超出了 integer 的范围，将会被解释为 float。同样，如果执行的运算结果超出了 integer 范围，也会返回 float。

3．浮点型

浮点数据类型可以用来存储数字，也可以用来保存小数。它提供了比整数大得多的精度。变量具有的确切精度和大小取决于使用 PHP 所依据的平台。在 32 位的操作系统中，它的有效范围是 1.7E-308 到 1.7E+308。给浮点数赋值，也有多种方法。

实例 2-12　为浮点数赋值

本实例为浮点型变量赋值，如代码 2-12 所示。

【代码 2-12】　float.php

```
<?php
    $float_1=900000000000;
    echo($float_1);
    echo("<br>");
    $float_2=9E10;
    echo($float_2);
    echo("<br>");
    $float_3=9E+10;
    echo($float_3);
?>
```

代码的执行结果如图 2-7 所示。



图 2-7　为浮点数赋值

4．字符串

字符串即连续的字符序列，字符串中的每个字符只占用一个字节。字符串在每种编程语

言中都得到了广泛的应用。

在 PHP 中，定义字符串，有以下 3 种方式。

（1）单引号方式。单引号字符串的赋值方法如下：

```
$Single_Str='我被单引号括起来了!';
```

如果要将字符串输出到浏览器中，可以使用关键字“echo”或者“print”，如下所示：

```
echo $Single_Str
```

或者：

```
print $Single_Str
```

但是如果想在浏览器中输出单引号，则必须在字符串中加入转义符号“\”，如在单引号前面加上转义符即“\'”即可输出单引号。

实例 2-13　单引号方式给字符串赋值

本实例演示了如何通过单引号给字符串赋值，如代码 2-13 所示。

【代码 2-13】　squote.php

```
<?
    $single_str='我被单引号括起来了!<br>';
    echo $single_str;
    $single_str='输出单引号：  \'嘿嘿，我在单引号里面\'<br>';
    echo $single_str;
    $single_str='输出双引号："我在双引号里面"<br>';
    print $single_str;
    $single_str='输双美元符号：$';
    print $single_str;
?>
```

代码执行结果如图 2-8 所示。

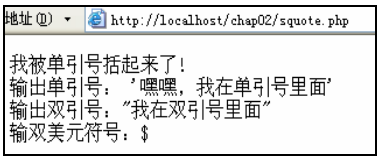


图 2-8　单引号应用

提示　在单引号内，允许的惟一转义符就是单引号“ ' ”。

（2）双引号方式。双引号的字符串的赋值方法如下：

```
$Double_Str="我被双引号括起来了!"
```

同理，要输出被双引号括起来的字符串，可以使用“echo”或者“print”关键字。为了在字符串中输出各种特殊字符，在双引号字符串中也可以使用转义符。

如要输出美元符号“$”或者反斜杠“\”可以使用下面语句：

```
$ Double_Str="输出美元符号：\$";
$ Double_Str="输出反斜杠：\\";
```

实例 2-14　双引号方式给字符串赋值

本实例演示了如何通过双引号方式给字符串赋值，如代码 2-14 所示。

【代码 2-14】 dquote.php

```
<?
    $Double_str="我被双引号括起来了!<br>";
    echo $single_str;
    $single_str="输出单引号：'嘿嘿，我在单引号里面'<br>"; //不需要转义符
    echo $single_str;
    $single_str="输出双引号：\"我在双引号里面\"<br>";  //需要转义符
    print $single_str;
    $single_str="输出美元符号：\$ <br>";  //转义或者不转义都可
    print $single_str;
    $single_str="输出反斜杠 ： \\ <br>";  //需要转义符
    print $single_str;
?>
```

代码执行结果如图 2-9 所示。

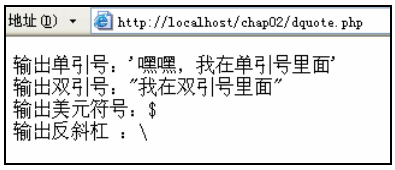


图 2-9 双引号应用

（3）Heredoc 方式。Heredoc 的字面意思是“此处文档”，用来提供一种较容易的长串管理方法。通常情况下，使用单引号或者双引号已经够用，但是在处理长字符串的时候，使用 Heredoc 方式更方便。尤其在从文件或者数据库中将大段文档输出到浏览器的时候，采用这种方法就非常方便了。

实例 2-15 Heredoc 方式给字符串赋值

本实例演示了使用 Heredoc 方式给字符串赋值，如代码 2-15 所示。

【代码 2-15】 Heredoc.php

```
<?
$heredoc_str =<<<heredoc_mark
    你好<br>
    美元符号  $ <br>
    反斜杠   \<br>
    "我爱你"<br>
    '我恨你'
heredoc_mark;
    echo $heredoc_str;
?>
```

代码执行结果如图 2-10 所示。



图 2-10 代码 2-10 的执行结果

从代码中可以看出，Heredoc 方式是将要输出的文档内容包括在“<<<heredoc_mark”和“heredoc_mark;”之间。

提示　“heredoc_mark;”表示字符串的结束，必须放在行首，不能缩进。另外“heredoc_mark”可以自己命名。

（4）串中的变量。在用双引号和 Heredoc 语法定义的串内可以包含其他变量。在 PHP 中，有两种办法可以包含其他变量：直接将变量名插入到字符串中，这种方法比较简单；将变量用大括号括起来，以便得到正确的结果。

实例 2-16　字符串中包含变量

本实例演示了字符串中包含变量名称的各种输出情况，如代码 2-16 所示。

【代码 2-16】　str_var.php

```
<?php
    $str_1 = "我是变量的值!";
    $str_2 = "str_1 : $str_1<br>";  //双引号字符串中包含变量$str_1
    echo $str_2;

    $str_1 = '我是变量的值';
    $str_2 = 'str_1 : $str_1<br>';  //单引号中包含字符串的值
    echo $str_2;

    $str_1 = "我是变量的值!";
    $str_2 = "str_1 : $str_12 <br>";  //引用的变量名后，多了个字符 2 即$str_12
    echo $str_2;

    $str_1 = "我是变量的值!";
    $str_2 = "str_1 : ${str_1}2 <br>";  //引用的变量名后，多了个字符 2 即$str_12
    echo $str_2;
?>
```

代码执行结果如图 2-11 所示。

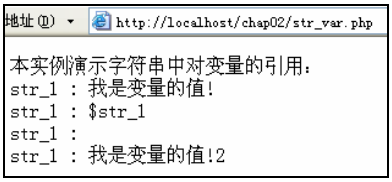


图 2-11　字符串中包含变量

从输出结果可以看出，如果是双引号字符串，字符串中包含的变量名称，输出的时候，会直接输出变量的值；而如果单引号字符串，字符串中包含的变量的名称，输出的时候，会直接输出变量的名称而不是变量的值。

另外从字符串“str_1 : $str_12
”看，如果变量名后面跟着非空格的字符，则会让编译器糊涂，编译器会自动地将“$str_12”认为是一个变量，而实际上并没有这个变量，因此会输出空值。为了解决这个问题，采用大括号的方式，即“str_1 : ${str_1}2
”，从图中可以

看到这样输出了正确的结果。

提示 ${str_1}和{$str_1}的效果是一样的。

关于字符串，在后面的章节中还有更加详细的说明。

5．复合数据类型

关于复合数据类型数组和对象，在后面会有专门的章节进行介绍。

6．特殊数据类型

PHP 中包含了两种特殊的数据类型，分别为资源和空值。

（1）资源。资源是 PHP 内的几个函数所需要的特殊数据类型，由编程人员来分配。如果编程人员没有对资源进行处理，就由 PHP 来处理无用资源。当使用了利用资源的函数时，常常有一个函数可用来处理变量的手工释放。

例如，创建外部资源函数是 xml_parser_create()函数。大多数 XML 函数要想正确工作，都需要对该资源进行引用。为了使用任一 XML 处理功能，必须创建这个 XML 解释器资源。无论一个变量是不是资源，它的处理都是后台进行的。

（2）空值。空值是最简单的数据类型。赋予了 NULL（空值）的任何一个值都不含有任何值。一个 NULL 值所表示的意思是说没有为该变量设置任何值。NULL 关键字是不区分大小写的。

关于特殊数据类型，由于使用频率不是很高，所以简单介绍到这里。

7．判断变量的数据类型

前面介绍了各种常用数据类型的简单应用。下面介绍如何判断一个变量的类型。在 PHP 中，提供了一个简单的函数来判断数据类型，该函数名称为 var_dump，函数的声明如下：

```
void var_dump ( mixed expression [, mixed expression [, ...]] )
```

其中，参数 expression 为变量名称或者表达式。

实例 2-17　判断变量的类型

本实例通过调用 var_dump 函数来判断一个变量的类型，如代码 2-17 所示。

【代码 2-17】　var_style.php

```
<?php
$a="";  //空字符串
$b = 3.1;  //浮点型
$c= TRUE;  //布尔型
$d =  2147483647;  //整型
$e =  2147483648;  //超过整型的范围，自动编程浮点型
$f= 0x80000000 ;   //十六进制
$g="hi";  //字符串
$h=array(1,2);  //数组

//分别输出类型
echo '$a 类型是：';
var_dump($a);
echo '<br>$b 类型是：';
```

```
var_dump($b);
echo '<br>$c 类型是：';
var_dump($c);
echo '<br>$d 类型是：';
var_dump($d);
echo '<br>$e 类型是：';
var_dump($e);
echo '<br>$f 类型是：';
var_dump($f);
echo '<br>$g 类型是：';
var_dump($g);
echo '<br>$h 类型是：';
var_dump($h);
?>
```

代码执行结果如图 2-12 所示。

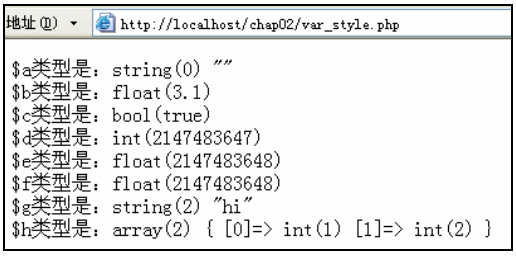


图 2-12　判断变量类型

8．通过 IS 函数判断数据类型

在 PHP 中，提供了很多的 IS 函数，用于判断是否是一个指定的对象或者变量，如 is_bool()（判断是否为布尔型）、is_float()（判断是否为浮点型）、is_int()（判断是否为整型）、is_numeric()（判断是否为数值型）、is_string()（判断是否为字符串）、is_array()（判断是否为数组）和 is_object()（判断是否为对象）。在编程中，可以根据需要来使用不同的函数。

实例 2-18　IS 函数判断变量类型

本实例将利用上面介绍的 is_bool、is_numeric 等函数判断数据类型，如代码 2-18 所示。

【代码 2-18】　is.php

```
<?php
$a=0.1;
$b='hi';
$c=True;
if (is_numeric($a))
    echo '$a 是数值型<br>';
else
    echo '$a 不是数值型<br>';

if (is_int($a))
    echo '$a 是整型<br>';
else
    echo '$a 不是整型<br>';
```

```
if (is_string($b))
    echo '$b 是字符串型<br>';
else
    echo '$b 不是字符串型<br>';

if (is_bool($c))
    echo '$c 是布尔型<br>';
else
    echo '$c 不是布尔型<br>';

?>
```

代码执行结果如图 2-13 所示。



图 2-13　判断变量类型

9．检查变量是否设置

如果 php.ini 文件的错误提示全部打开，即包括错误、警告等，可能是获取了一个没有赋值的变量。为了保证代码能够在各种设置下安全运行，最好在使用变量之前判断是否定义了该变量。这里介绍两个函数用来判断是否变量已经定义。

（1）isset()函数。该函数检测变量是否设置，声明如下所示：

```
bool isset ( mixed var [, mixed var [, ...]] )
```

如果 var 存在，则返回 TRUE，否则返回 FALSE。如果已经使用 unset()释放了一个变量，则该变量没有被设置。若使用 isset()测试一个被设置成 NULL 的变量，将返回 FALSE。同时要注意的是一个 NULL 字节（"\0"）并不等同于 PHP 的 NULL 常数。


提示　isset()只能用于变量，因为传递任何其他参数都将造成解析错误。若想检测常量是否已设置，可使用 defined()函数。

（2）Empty()函数。该函数检查一个变量是否为空，声明如下所示：

```
bool empty ( mixed var )
```

如果参数 var 是非空或非零的值，则 empty()返回 FALSE。换句话说，""、0、"0"、NULL、FALSE、array()、var $var;以及没有任何属性的对象都将被认为是空的，如果 var 为空，则返回 TRUE。


注意　虽然 Empty 和 isset 函数都可以判断一个变量是否设定。但是 Empty 和 isset 在对待 0 和空字符串的时候，结果是不一样的。空字符串和 0，对 Empty 来说，表示为空；而对 isset 函数来说，表示已经设置。

实例 2-19　检测变量是否设置

本实例检测变量是否已经设置，并比较 Empty 和 Isset 两个函数的区别，如代码 2-19 所示。

【代码 2-19】　dec_var.php

```
<?php
$a=0.1;    //定义变量
$b=0;    //0 值
$c="";  //空字符串
if (isset($a))
    echo '$a 已经被定义<br>';
else
    echo '$a 没有被定义<br>';

if (!empty($a))
    echo '$a 不为空<br>';
else
    echo '$a 为空<br>';

if (isset($b))
    echo '$b 已经被定义<br>';
else
    echo '$b 没有被定义<br>';

if (!empty($b))
    echo '$b 不为空<br>';
else
    echo '$b 为空<br>';


if (isset($c))
    echo '$c 已经被定义<br>';
else
    echo '$c 没有被定义<br>';

if (!empty($c))
    echo '$a 不为空<br>';
else
    echo '$a 为空<br>';

if (isset($d))
    echo '$d 已经被定义<br>';
else
    echo '$d 没有被定义<br>';

if (!empty($d))
    echo '$d 不为空<br>';
else
    echo '$d 为空<br>';
?>
```

代码执行结果如图 2-14 所示。

从执行结果可以看出，对于没有定义的变量，Empty 和 Isset 都可以进行判断。但是对于空字符串和 0 值，两个函数的判断结果不一致。

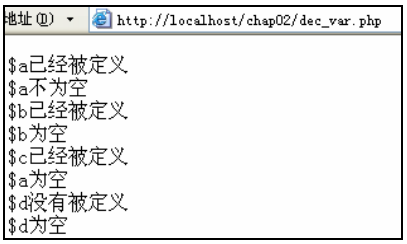


图 2-14　判断变量是否设置或者为空

2.2.3　数据类型转换

PHP 中处理类型转换的方法和其他语言类似。在类型转换中，明确地把一个变量从一种数据类型转换成另外一种数据类型。在 PHP 中，可以实现下列类型的转换，如表 2-1 所示。

表 2-1　　变量类型转换

作　用	数 据 类 型	作　用	数 据 类 型
把值转化成布尔值	(bool),(boolean)	把值转化为数据	(array)
把值转化成整数	(int),(integer)	把值转化为对象	(object)
把值转化成浮点数	(float),(double),(real)	把值转化成字符串	(string)

1．布尔转换

要明确地将一个值转换成 boolean，可以用（bool）或者（boolean）来强制转换。但是很多情况下不需要用强制转换，因为当运算符、函数或者流程控制需要一个 boolean 参数时，该值会被自动转换。

实例 2-20　布尔转换

本实例演示了如何将其他数值或者变量转换成布尔类型，如代码 2-20 所示。

【代码 2-20】　bool.php

```
<?php
    echo gettype((bool) "") . "<br>";      // bool(false)
    echo gettype((bool) 1)  . "<br>";       // bool(true)
    echo gettype((bool) -2) . "<br>";       // bool(true)
    echo gettype((bool) "foo") . "<br>";    // bool(true)
    echo gettype((bool) 2.3e5) . "<br>";    // bool(true)
    echo gettype((bool) array(12)) . "<br>"; // bool(true)
    echo gettype((bool) array()) . "<br>";  // bool(false)
?>
```

代码执行结果如图 2-15 所示。

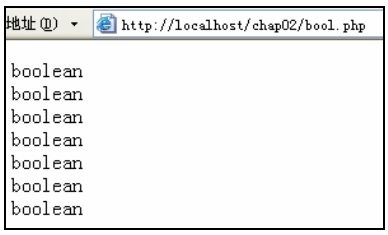


图 2-15　布尔转换

从执行结果可以看出，可以强制将多种数据类型转换成布尔类型。

2．字符串转换

可以用（string）标记或者 strval()函数将一个值转换为字符串。当某表达式需要字符串时，字符串的转换会在表达式范围内自动完成。例如当使用 echo()或者 print()函数时，或者将一个变量值与一个字符串进行比较的时候。

在进行字符串转换的时候，有以下规则。

（1）布尔值 True 将被转换为字符串"1"，而值 False 将被表示为""（即空字符串）。这样就可以随意地在布尔值和字符串之间进行比较。

（2）整数或浮点数数值转换成字符串时，字符串即为这些数值数字（浮点数还包含指数部分）。

（3）数组将被转换成字符串"Array"，因此无法通过 echo()或者 print()函数来输出数组的内容。

（4）对象将被转换成字符串"Object"。如果希望得到该对象所依附的类的名称，要使用函数 get_class()。

（5）资源类型将会以"Resource id #1"的格式被转换成字符串，其中 1 是 PHP 在运行时给资源指定的惟一标识。如果希望获取资源的类型，要使用函数 get_resource_type()。

（6）NULL 将被转换成空字符串。



提示　将数组、对象或者资源打印出来，并不能提供任何关于这些值本身的有用的信息。可以使用函数 print_r()和 var_dump()，对于调试来说，这些是更好的打印值的方法。



提示　可以将 PHP 的值转换为字符串以永久地储存它们。这种方法被称为串行化，可以用函数 serialize()来完成该操作。如果安装 PHP 时建立了 WDDX 支持，还可以将 PHP 的值串行化为 XML 结构。

如果要将字符串转化成数值，要遵循下面的原则。

（1）如果包括“.”，“e”或“E”或其中任何一个字符的话，字符串被当作 float 来求值。否则就被当作整数。

（2）该值由字符串最前面的部分决定。如果字符串以合法的数字数据开始，就用该数字作为其值，否则其值为 0（零）。合法数字数据由可选的正负号开始，后面跟着一个或多个数字（可选地包括十进制分数），后面跟着可选的指数。指数是一个“e”或者“E”后面跟着一个或多个数字。

实例 2-21　字符串和数值之间的转换

本实例演示了字符串转化成数值以及数值转化成字符串的几种情况，如代码 2-21 所示。

【代码 2-21】　change_style.php

```
<?php
$a=123456;
$b=123.456;
$c=-1.3e8;
```

```
//通过 echo 函数输出时，自动转换成
echo $a . "abc<br>";
echo $b ."cde<br>";
echo $c . "abc<br>";

//字符串转换成数值
$foo = 1 + "10.5";
echo $foo . "<br>";
$foo = 1 + "-1.3e3";
echo $foo ."<br>";
$foo = 1 + "bob-1.3e3";
echo $foo ."<br>";
$foo = 1 + "bob3";
echo $foo ."<br>";
$foo = 1 + "10 Small Pigs";
echo $foo ."<br>";
$foo = 4 + "10.2 Little Piggies";
echo $foo ."<br>";
$foo = "10.0 pigs " + 1;
echo $foo ."<br>";
$foo = "10.0 pigs " + 1.0;
echo $foo ."<br>";
$foo = 1 + "010";
echo $foo ."<br>";
$foo = 1 + "0x10";
echo $foo ."<br>";
$foo = 1 + "\x10";
echo $foo ."<br>";
$bar = 1 + "\x35"; //0x35 is ASCII for '5'
echo $foo ."<br>";
$foo = 1 + "0x\x41\x31"; //0x41 is ASCII 'A'; 0x31 is '1'
echo $foo ."<br>";
?>
```

执行结果如图 2-16 所示。

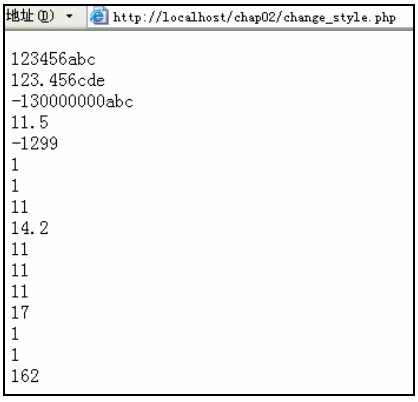


图 2-16　字符串和数值转换

3．整型转换

要将一个值转换为 integer，用（int）或（integer）强制转换。不过大多数情况下都不需

要强制转换，因为当运算符、函数或流程控制需要一个 integer 参数时，值会自动转换。还可以通过函数 intval()将一个值转换成整型。

下面介绍将各种其他数据类型转换成整型的方法。

（1）从布尔值转换，False 将产生出 0，True 将产生出 1。

（2）从浮点数转换。当从浮点数转换成整数时，数字将被取整（丢弃小数位）。如果浮点数超出了整数范围（通常为+/–2.15e+9 = 2^31），则结果不确定，因为没有足够的精度使浮点数给出一个确切的整数结果。

提示　不要将未知的分数强制转换为 integer，这样会导致意外的结果。

2.2.4　PHP 中的预定义变量

PHP 中有一些可供利用的预定义变量。作为一种良好的编程习惯，应该避免使用任何与预定义变量同名的变量。

编程人员可以很方便的在程序中访问 PHP 中的预定义变量。

下面简单列举几个常用变量。

REQUEST_METHOD：访问页面时的请求方法。例如“GET”、“HEAD”、“POST”和“PUT”。

REQUEST_TIME：请求开始时的时间戳。

QUERY_STRING：查询（query）的字符串（URL 中第一个问号?之后的内容）。

DOCUMENT_ROOT：当前运行脚本所在的文档根目录。在服务器配置文件中定义。

HTTP_ACCEPT：当前请求的 Accept 头信息的内容。

HTTP_ACCEPT_CHARSET：当前请求的 Accept-Charset 头信息的内容。

SERVER_PROTOCOL：请求页面时通信协议的名称和版本，例如，“HTTP/1.0”。

REQUEST_METHOD：访问页面时的请求方法。例如“GET”、“HEAD”、“POST”和“PUT”。

下面介绍访问预定义变量的方法。

（1）在 PHP 中可以通过 3 种方式来访问这些预定义变量。最原始的访问方法就是直接在全局变量前面加美元符号来访问，如下所示：

```
$REQUEST_METHOD
```

但是在 PHP5 中这种直接的访问方式已经被屏蔽了，所以要想通过这种方法访问预定义变量，必须修改 php.ini 的设置。

将“register_globals = OFF”改成“register_globals = ON”，然后重新启动 Apache 服务器。

实例 2-22　使用“$”访问预定义变量

本实例将通过“$”直接加预定义变量名称的方法访问预定义变量的值，如代码 2-22 所示。

【代码 2-22】　$_var.php

```
<?php
echo '提交方法为:'. $REQUEST_METHOD;
echo '<br>网络协议为:' .$SERVER_PROTOCOL;
?>
```

代码执行结果如图 2-17 所示。



图 2-17　美元符号访问预定义变量

这种方法容易给编程人员带来一些不必要的麻烦，因此 PHP 默认是不支持这种访问方式的。当然读者应该掌握和了解，尤其是阅读以前版本的 PHP 源码的时候。

（2）用$HTTP_*_VARS 方法访问预定义变量。$HTTP_*_VARS 是一种长格式的访问方式。其中“*”根据不同的变量使用不同的内容。通常这些预定义变量可以分成以下几种。

服务器变量：系统的服务器设置变量（访问方式$HTTP_SERVER_VARS）。

环境变量：系统的环境设置变量（访问方式$HTTP_ENV_VARS）。

Cookies 变量：Cookies 变量（访问方式$HTTP_Cookies_VARS）。

GET 变量：通过 GET 方法请求变量（访问方式$HTTP_GET_VARS）。

POST 变量：通过 POST 方法请求变量（访问方式$HTTP_POST_VARS）。

REQUEST 变量：通过表单请求变量（访问方式$HTTP_REQUEST_VARS）。

Session 变量：会话控制变量（访问方式$HTTP_SESSION_VARS）。

文件上传变量：用户将文件传到服务器的变量（访问方式$HTTP_POST_FILES）。

在 PHP4.0 之后，这种访问方式就被禁止了，因此为了启用这种访问方式，必须修改 php.ini 文件的设置。

将“register_long_arrays = OFF”改成“register_long_arrays = ON”，然后重新启动 Apache 服务器。

实例 2-23　通过$HTTP_*_VARS 访问预定义变量

本实例通过$HTTP_*_VARS 访问预定义变量，如代码 2-23 所示。

【代码 2-23】　http_vars.php

```
<?php
echo '主机名称:'.$HTTP_SERVER_VARS['HTTP_HOST'];
echo '<br>服务器根路径:'.$HTTP_SERVER_VARS['DOCUMENT_ROOT'];
?>
```

执行结果如图 2-18 所示。



图 2-18　http_*_var 方法访问预定义变量

（3）超全局变量访问预定义变量。在 PHP4 之后，采用了一种新的方式来访问预定义变量。系统提供了一些定义好的超全局变量数组，用户可以直接调用这些数组进行访问。

PHP 中定义了以下超全局变量。

$GLOBALS：包含一个引用指向每个当前脚本的全局范围内有效的变量。该数组的键名为全局变量的名称。

$_SERVER：变量由 Web 服务器设定或者直接与当前脚本的执行环境相关联。类似于

$HTTP_SERVER_VARS 数组。

$_GET：经由 URL 请求提交至脚本的变量。类似于$HTTP_GET_VARS 数组。

$_POST：经由 HTTP POST 方法提交至脚本的变量。类似于$HTTP_POST_VARS 数组。

$_COOKIE：经由 HTTP Cookies 方法提交至脚本的变量。类似于$HTTP_COOKIE_VARS 数组。

$_FILES：经由 HTTP POST 文件上传而提交至脚本的变量。类似于$HTTP_POST_FILES 数组。

$_ENV：执行环境提交至脚本的变量。类似于$HTTP_ENV_VARS 数组。

$_REQUEST：经由 GET、POST 和 COOKIE 机制提交至脚本的变量，因此该数组并不值得信任。所有包含在该数组中的变量的存在与否以及变量的顺序均按照 php.ini 中的 variables_order 配置指示来定义。

$_SESSION：当前注册给脚本会话的变量。类似于$HTTP_SESSION_VARS 数组。

实例 2-24　使用超全局访问预定义变量

本实例将介绍如何通过超全局变量访问预定义变量，这也是 PHP 推荐的访问方式，如代码 2-24 所示。

【代码 2-24】　super_vars.php

```
<?php
echo '当前执行脚本文件:'.$_SERVER['PHP_SELF'];
echo '<br>当前执行脚本文件绝对路径:'.$_SERVER['SCRIPT_FILENAME'];
?>
```

执行结果如图 2-19 所示。

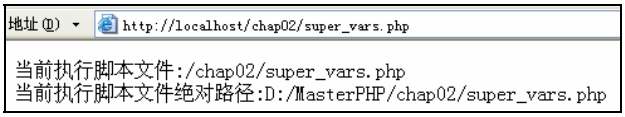


图 2-19　超全局变量

2.2.5　变量的引用

如果声明了一个变量，并且又让该变量引用了另外一个变量，那么这个刚刚建立的变量将指向另外的那个变量。这意味着对这两个变量的任意一个所做的修改都改变这两个变量中所存储的值。引用是一种通过多个标志符参考一个变量的方法。

要想使一个变量成为一个引用，必须使用“与”字符“&”。

实例 2-25　变量的引用

本实例介绍如何在 PHP 中进行变量的引用，如代码 2-25 所示。

【代码 2-25】　ref_var.php

```
<?php
    $myName="little";
    $YourName=&$myName;  //引用变量
```

```
    $myName="littlePig<br>";

    print($YourName);

    $YourName="littleCat<br>";
    print($myName);
?>
```

执行结果如图 2-20 所示。



图 2-20　变量的引用

在本例中，当把$youName 设置成&myName 时，创建了一个指向$myName 的引用。无论修改$myName 还是修改$yourName 的值，这两个变量的值都会改变。

在后面学习函数的时候会发现，通过引用变量进行参数传递很方便。

2.2.6　变量的变量

变量的变量是指该变量的名称由另外一个变量的值来确定。也许听起来会糊涂，下面通过一个简单的实例说明。

实例 2-26　变量的变量

本实例介绍 PHP 中变量的变量的应用，如代码 2-26 所示。

【代码 2-26】　var_var.php

```
<?php
    $myStr="myName";
    $$myStr="Marry";
    echo $myStr;
    echo "<br>";
    echo $$myStr;
    $myStr="yourName";
    echo $$myStr;
?>
```

执行结果如图 2-21 所示。



图 2-21　变量的变量

在上面的代码中，语句“$$myStr="Marry";”相当于“$myName="Marry";”，因为$myStr 的值为“myName”，这样就可以动态地创建变量了。因为变量名是动态创建的，因此当更换了变量名之后，需要重新赋值。如将$myStr 改成“yourName”，则$$myStr 变成“$yourName”，而该变量并没有赋值，所以最后一句输出$$myStr 的结果并不是“Marry”。

2.2.7　常量

在 PHP 中，一个常量可以理解为值不变的变量。比如需要经常使用圆周率、重力系数等一般不会变化的值时，通常会将这些值保存在常量中，便于代码的维护和修改。

1．自定义常量

常量通常具有如下属性。

（1）常量是区分大小写的，可以是大写字母也可以是小写字母，但是在定义的时候可以指定是否大小写敏感。

（2）命名规则和变量的规则一样。

（3）常量的开头只能是字母或者下划线，后面可以接任意个数的字母字符。

（4）默认的情况下，它为全局变量，即在当前脚本的任何地方都可以使用。

（5）一旦定义了常量之后，在其他地方不得再次定义名称相同的常量或者变量。

（6）在定义常量名称的时候，尽量全部使用大写字符，这样容易阅读和识别。

在定义常量时必须使用 define 函数。通常可以通过该函数定义 4 种类型的值：字符串、布尔值、浮点数或者整数。define 函数的语法如下：

```
bool define ( string name, mixed value [, bool case_insensitive] )
```

define 函数需要 3 个参数，第一个参数为常量名称，即标志符，第二个参数为常量的值，第三个参数指定是否大小写敏感，如果设定为 True，表示不敏感。

可以简单的通过指定其名字来取得常量的值，不要在常量前面加上$符号。当需要在程序中动态获取常量值的时候，可以使用 constant 函数。constant 函数要求一个字符串作为参数，并返回该常数的值。

要判断一个常量是否已定义，可以使用 defined 函数，该函数也需要一个字符串参数，该参数为需要检测的常量名称。如果成功，则返回 True。

还可以通过 get_defined_constants()函数来获取所有当前已经定义的常数列表。

实例 2-27　常量的定义

本实例介绍如何定义常量，如代码 2-27 所示，文件名为 define_var.php。

【代码 2-27】　define_var.php

```
<?php
    define("CONSTANT", "Hello world\n");
    echo CONSTANT . "<BR>"; // 输出 "Hello world."
    echo Constant . "<BR>"; // 输出 "Constant" ，表示没有该常量

    define("GREETING", "Hello you\n", true);
    echo GREETING. "<BR>"; // 输出"Hello you."
    echo Greeting. "<BR>"; // 输出 "Hello you."，因为设定大小写不敏感

    define("MAXSIZE", "100\n");
    echo MAXSIZE;  //输出 100
    echo constant("MAXSIZE"). "<BR>"; // 输出 100

    echo(defined("ONE")). "<BR>";    //未定义，因此输出空行
```

```
    define("ONE","Hello,One");
    echo(defined("ONE")). "<BR>";  //如果定义返回 True，使用 echo 输出显示 1
?>
```

执行结果如图 2-22 所示。



图 2-22　常量的定义

2．预定义常量

正如 PHP 中有预定义的变量一样，也有预定义的常量。可以使用这些常量获取 PHP 中的信息。

PHP 中定义了以下一些常量。

（1）__FILE__：这个默认常量是 PHP 程序文件名。若引用文件（include 或 require）则在引用文件内的该常量为引用文件名，而不是引用它的文件名。

（2）__LINE__：这个默认常量是 PHP 程序行数。若引用文件（include 或 require）则在引用文件内的该常量为引用文件的行，而不是引用它的文件行。

（3）PHP_VERSION：这个内建常量是 PHP 程序的版本，如'3.0.8-dev'。

（4）PHP_OS：这个内建常量指执行 PHP 解析器的操作系统名称，如'Linux'。

（5）TRUE：这个常量就是真值（true）。

（6）FALSE：这个常量就是假值（false）。

（7）NULL：一个 Null 值。

（8）E_ERROR：这个常量指到最近的错误处。

（9）E_WARNING：这个常量指到最近的警告处。

（10）E_PARSE：本常量为解析语法有潜在问题处。

（11）E_NOTICE：这个常量为发生不寻常但不一定是错误处。

提示　这些 E_开头的常量，可以参考 error_reporting()函数。

实例 2-28　输出预定义常量

本实例输出一些预定义的常量，如代码 2-28 所示，文件名为 pre_var.php。

【代码 2-28】　pre_var.php

```
<?php
echo "程序当前行：". __LINE__;
echo "<br>操作系统：".PHP_OS;
echo "<br>PHP 版本：" .PHP_VERSION
?>
```

执行结果如图 2-23 所示。

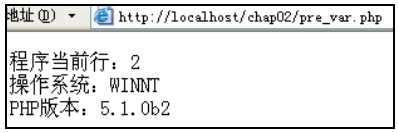


图 2-23　输出预定义变量

2.3　PHP 的运算符

运算符是一个特殊符号，它对一个值或一组值执行一个指定的操作。PHP 具有 C，C++ 和 Java 语言中常见的运算符，这些运算符的优先权也是一致的。

在 PHP 中包含以下运算符：

（1）算术运算符；
（2）逻辑运算符；
（3）比较运算符；
（4）位运算符；
（5）赋值运算符；
（6）字符串运算符；
（7）执行运算符；
（8）加一/减一运算符；
（9）错误控制运算符。

下面分别介绍各种常用的运算符。

2.3.1　算术运算符

算术运算（Arithmetic operators）符号，是用来处理四则运算的符号，是最简单，也是最常用的符号，尤其是数字的处理，几乎都会使用到算术运算符号。

常用的算术运算符如表 2-2 所示。

表 2-2　常用的算术运算符

符　号	示　例	符　号	示　例
+	加法运算：$a + $b	/	除法运算：$a / $b
−	减法运算：$a – $b	%	取余数运算：$a % $b
*	乘法运算：$a * $b		



除号（“/”）总是返回浮点数，即使两个运算数是整数（或由字符串转换成的整数）也是一样。

实例 2-29　算术运算符

本实例介绍常用算术运算符的应用，如代码 2-29 所示。

【代码 2-29】　arithmetic.php

```
<?php
    $ONE= 1;
```

```
    $TWO= 2;
    $TREE= 3;
    echo $ONE+$TWO . "<br>";  //加
    echo $ONE-$TWO . "<br>";  //减
    echo $ONE*$TWO . "<br>";  //乘
    echo $ONE/$TWO . "<br>";  //除
    echo $TREE%$TWO. "<br>";  //取余数
?>
```

执行结果如图 2-24 所示。

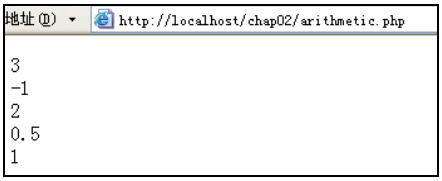


图 2-24　算术运算符应用



代码中的“.”表示连接两个字符串。在后面章节会专门介绍。“
”是 HTML 标记，表示换行。

2.3.2　赋值运算符

在做简单的操作时，赋值运算符提供了把值赋给变量的一个快捷方法。读者肯定对赋值运算符“=”非常熟悉。但是在 PHP 中，还提供了若干组合赋值运算符。这些赋值运算符提供了做基本算术和串运算的方法。

在 PHP 中，具体的赋值运算符如表 2-3 所示。

表 2-3　　赋值运算符

操　作	运 算 符	示　例	展 开 形 式
赋值	=	$a =3	$a=3
加	+ =	$a + =3	$a=$a + 3
减	– =	$a – =3	$a=$a – 3
乘	* =	$a * =3	$a=$a * 3
除	/ =	$a / =3	$a=$a / 3
连接字符串	. =	$a . ="3"	$a=$a . "3"
取余数	% =	$a % =3	$a=$a % 3
位 AND	&	$a & =3	$a=$a & 3
位包含 OR	\| =	$a \| =3	$a=$a \| 3
位排斥 OR（XOR）	^ =	$a ^ =3	$a=$a ^ 3
位 NOT	~ =	$a ~ =3	$a=$a ~ 3
位左移	<< =	$a << =3	$a=$a << 3
位右移	>> =	$a >> =3	$a=$a >>3

实例 2-30　赋值运算符

本实例介绍各种常用的赋值运算符，如代码 2-30 所示。

【代码 2-30】　evaluate.php

```
<?php
    $a= 2;
    $a +=3;  //$a=$a+3，值为 5
    echo $a . "<br>";
    $a -=3;   //$a=$a-3，值为 2
    echo $a . "<br>";
    $a *=3;   //$a=$a*3，值为 6
    echo $a . "<br>";
    $a /=3;   //$a=$a/3，值为 2
    echo $a . "<br>";
?>
```

执行结果如图 2-25 所示。



图 2-25　赋值运算符

2.3.3　位逻辑运算符

位运算是指对二进制位从低位到高位对齐后进行运算操作，如检测、移位等。PHP 语言所提供的位运算符如表 2-4 所示。

表 2-4　　位运算符

位 运 算 符	作　用	示　例	位 运 算 符	作　用	示　例
&	按位与	$a & $b	~	按位取反	~$a
\|	按位或	$a \| $b	<<	向左移位	$a << $b
^	按位异或	$a ^ $b	>>	向右移位	$a >> $b

若变量为实数类型，PHP 会自动取整，然后再进行位运算。

下面举几个按位运算的例子。

（1）按位与：7&14。

7 的二进制表示为：　00000111

14 的二进制表示为：00001110

7	00000111
14	00001110
按位与（&）	00000110

所以，7&14 的结果为 6。

（2）按位或：7|14。

7	00000111
14	00001110
按位或（\|）	00001111

所以，7|14 的结果为 15。

（3）按位异或：7^14。

7	00000111
14	00001110
按位或（^）	00001001

所以，7^14 的结果为 9。

实例 2-31　位运算符

本实例介绍位逻辑运算符的应用，如代码 2-31 所示。

【代码 2-31】　bit.php

```
<?php
    $a= 7;
    $b= 14;
    $myVal= $a & $b;  //位与
    echo  $myVal. "<br>";
    $myVal=$a | $b;  //位或
    echo  $myVal. "<br>";
    $myVal= $a ^ $b;  //位异或
    echo  $myVal. "<br>";
    $myVal= ~$a;  //为取反
    echo  $myVal. "<br>";
?>
```

执行结果如图 2-26 所示。



图 2-26　位运算符

2.3.4　三元运算符

一般的运算符都是二元运算符，而在 PHP 中有三元运算符。所谓三元操作符，就是在表达式中有 3 个操作数。三元运算符是从 PHP 的鼻祖 C 语言继承而来的。

三元运算符的使用形式为：表达式 1?表达式 2：表达式 3。

三元运算符能够提供简单的逻辑判断，但是建议读者少使用，因为它会导致非常处理和引起混淆的代码。

三元表达式中的第一个操作数是要计算的测试表达式，在问号和冒号之间的第二个表达式只有在测试表达式的计算结果为 True 的时候才会计算。出现在冒号后面的最后一个表达式只有在测试表达式的计算结果为 False 的时候才会计算。

三元运算符的应用在 2.3.4 小节的比较运算符中介绍。

2.3.5　比较运算符

PHP 语言中的比较运算符如表 2-5 所示。

表 2-5　比较运算符

比较运算符	作　用
<	小于
>	大于
<=	小于或等于
>=	大于或等于
==	等于，两个变量的值相同
!=	不等于
===	等价，如果一个变量的值等于另外一个变量的值，同时数据类型也相同

前 4 种关系运算符的优先级相同，后两种的优先级也是相同的，但前 4 种运算符的优先级高于后两种。使用关系运算符运算的结果只有“真”（1）或“假”（0）。

比较运算符比较容易理解，下面通过实例学习。

实例 2-32　比较运算符

本实例演示了如何使用比较运算符，如代码 2-32 所示。

【代码 2-32】　compare.php

```
<?php
    echo("10 小于 10 吗：" . ((10<10)?"True":"False") . "<br>");
    echo("10 小于等于 10 吗：" . ((10<=10)?"True":"False") . "<br>");
    echo("10 大于 10 吗：" . ((10>10)?"True":"False") . "<br>");
    echo("10 大于等于 10 吗：" . ((10>=10)?"True":"False") . "<br>");
    echo("10 等于 10 吗：" . ((10==10)?"True":"False") . "<br>");
    echo("10 不等于 10 吗：" . ((10!=10)?"True":"False") . "<br>");
?>
```

执行结果如图 2-27 所示。

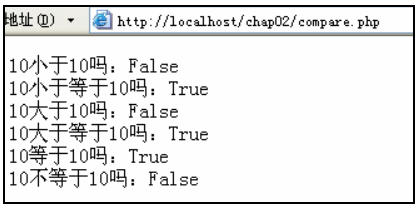


图 2-27　比较运算符

2.3.6　字符串运算符

字符串运算（String operator）的运算符号只有一个，即英文的句号“.”。它可以将字符串连接起来，变成合并的新字符串。当然还有一种简单的写法，即“.=”，这在赋值运算符中有介绍。

实例 2-33　字符串运算符

本实例介绍常用的字符串运算符，如代码 2-33 所示。

【代码 2-33】 string.php

```
<?php
    $a="Hello ";
    $b="World";
    $ab=$a . $b ."<br>";   //字符串连接符号.
    echo $ab;

    $a="some";
    $a .="thing" . "<BR>";  //赋值运算符.=
    echo $a;

    $a="24 five 6" + 6;
    echo $a . "<BR>";

    $a="24five 6" + 6;
    echo $a . "<BR>";

    $a="five 6" + 6;
    echo $a . "<BR>";

    $a="24 five" . 6;
    echo $a . "<BR>";

    $a="24five" . 6;
    echo $a . "<BR>";
?>
```

执行结果如图 2-28 所示。



图 2-28 字符串运算符

2.3.7 递增和递减运算符

在前面学习了算术运算符，来增加变量的值。这是最常见的做法，但是它适合有两个或者以上不同操作数的场合。在只有一个操作数的情况下，由于变量重复了两次，表达式冗长，所以没有必要。例如$Num=$Num+1 或者$Num=$Num-1。可以不使用数学运算符，而利用++或者--运算符，它们也称为一元运算符。

一元运算符有两种使用方法，一种是将运算符放在变量的前面，这种使用方法通常是先将变量的值增加 1 或者减少 1，然后返回变量的新值。而另一种方法是将运算符放在变量的后面，这种方法是先返回变量的当前值，然后变量的当前值增加 1 或者减少 1。

表 2-6 所示列出了一元运算符的常用运算规则。

表 2-6　一元运算符的规则

运　算　符	作　　用	说　　明
++$myVar	先加	$myVar 的当前值增加 1，然后返回新值
$myVar++	后加	先返回$myVar 的当前值，然后$myVar 的当前值增加 1
—$myVar	先减	$myVar 的当前值减少 1，然后返回新值
$myVar—	后减	先返回$myVar 的当前值，然后$myVar 的当前值减少 1

实例 2-34　递增和递减运算符

本实例介绍递增和递减运算符的应用，如代码 2-34 所示。

【代码 2-34】　plus.php

```
<?php
    $a=20;
    echo "a++:" . $a++ ."<br>";  //后加
    echo "变量 a 的新值:" . $a ."<br>";

    $a=20;  //重新赋值
    echo "++a:" . ++$a ."<br>";  //先加
    echo "变量 a 的新值:" . $a ."<br>";


    $a=20;  //重新赋值
    echo "a--:" . $a-- ."<br>";  //后减
    echo "变量 a 的新值:" . $a ."<br>";

    $a=20;  //重新赋值
    echo "--a:" . --$a ."<br>";  //先减
    echo "变量 a 的新值:" . $a ."<br>";
?>
```

执行结果如图 2-29 所示。



图 2-29　递增和递减运算符

2.3.8　逻辑运算符

逻辑运算符是程序设计中不可缺少的一组运算符，如表 2-7 所示，它对程序中的抉择十分有用。



AND 和 OR 逻辑运算符都是短路运算符，即一旦该表达式的要求得到满足，它就会停止计算。

表 2-7 逻辑运算符的规则

逻辑运算符	作　用	说　明
And	逻辑与	$a and $b：如果$a 与$b 都为 TRUE，则返回 True
Or	逻辑或	$a or $b：如果$a 或$b 任一为 TRUE，则返回 True
Xor	逻辑异或	$a xor $b：如果$a 或$b 任一为 TRUE，但不同时是，则返回 True
Not	逻辑非	! $a：如果$a 不为 TRUE，则返回 True
And	逻辑与	$a && $b：如果$a 与$b 都为 TRUE，则返回 True
Or	逻辑或	$a \|\| $b：如果$a 或$b 任一为 TRUE，则返回 True

如下代码：

```
(False && ($a > 10))
```

这条语句，永远不会实际计算右边的表达式（$a>10），因为，无论（$a>10）是否成立，整个表达式永远是为 Fasle。PHP 从左向右计算表达式，因此一旦遇到 Fasle &&，解释程序就自动将整个表达式的值设置为 Fasle。

同理，如下代码：

```
(True || ($a >10))
```

代码中（$a>10）永远不会被执行，因为当 PHP 解释程序遇到 True ||就会自动将整个表达式设置为 True。

2.3.9 运算符优先级

PHP 中的运算符优先顺序可以参考表 2-8，在混合式的情形下，愈往下表示优先权愈高。

表 2-8 运算符优先级

结合方向	运　算　符	结合方向	运　算　符
左	,	左	^
左	or	左	&
左	xor	无	= = != = = = ! = =
左	and	无	< < = > > =
右	print	左	<< >>
右	= += -= *= /= .= %= &= \|= ^= ~= <<= >>=	左	+ - .
左	? :	左	* / %
左	\|\|	右	! ~ ++ -- (int) (float) (string) (array) (object) @
左	&&	右	[
左	\|	无	new



尽管!比=的优先级高，PHP 仍旧允许类似如下的表达式 “if (!$a = foo())”，在此例中，foo()的输出被赋给了$a。

2.4 表　达　式

表达式是 PHP 最重要的基石。在 PHP 中，几乎所写的任何形式都是一个表达式。最基本的表达式形式是常量和变量。如 “$a =10”，即表示将值 10 分配给变量$a。

表达式的概念十分抽象，但表达式有一个具体的代码表示。一般说来，一个表达式是任何一个有界的（即有边界限制或者标记的）符号集合。这些符号只是一些对 PHP 解释程序由

具体含义的原子单元。它们可以是变量名、函数名、运算符、字符串值、数值和括号等。

例如下面的代码包含了一个无用的表达式：

```
<?php
    "foo";
?>
```

上述表达式实际上不做任何事情，但它仍然是一个单个符号所组成的有效表达式。下面的代码是由两个表达式组成的比较复杂的脚本：

```
<?php
    "fine";
    $a="hello";
?>
```

实际上，也可以将上面的代码说成 4 个表达式，即“fine”、$a、“hello”和$a="hello"，但是这样并没有什么意义。在上面的代码中，PHP 代码并没有进行有意义的操作。因为表达式并不是都是有效的，只要能够让 PHP 语言解释程序能够辨认它们，然后由解释程序尽力翻译它们的含义。

通常的情况下，在 PHP 代码中，使用一个分号来区分表达式，但是表达式可以包含在括号内。

广义上来解释，一条 PHP 语句，就是一个表达式后面跟着一个分号。实际上，编程人员可以用表达式来做任何事情，比如调用一个内部函数、编写一个控制语句、给变量赋值等。

运算符优先级决定了表达式的计算顺序。除了运算符优先级之外，运算符的结合原则对表达式的计算也有影响。若某些低优先级的运算符需要优先计算，必须使用括号。

如下面代码所示：

```
$a=10+9*20
```

在上面的表达式中，PHP 解释器会首先计算 9 乘以 20 的值，然后将该值与 10 相加。但是如果想首先计算 10 加 9 的值，然后再与 20 相乘，则可以使用下面的语句：

```
$a=(10+9)*20
```



如果编写逻辑表达式的时候，对运算符的优先级不清楚，为了防止出错，可以增加括号来确保正确。

2.5　控 制 语 句

由于 PHP 的大部分语法都是继承了 C 语言的语法，因此，在流程控制方面，也是有着和 C 语言极其类似的循环。PHP 的循环不像 ASP 可以使用 goto 的 BASIC 语法，PHP 是结构化的程序语言，流程的设计上有一定的规定，而不能用 BASIC 的概念跳到别的部分中。

流程的部分分隔符号都是使用“{”当作部分的开头，用“}”当作结尾，和 C 语言相同。而 PHP 语法中每条指令结束时都要加上分号“;”，但是在部分结尾符号“}”后面不用加上分号结束。

2.5.1　条件语句

有时候需要测试一个条件，并且根据条件返回的结果采取对应的措施。可以使用条件语句来实现这个任务。在条件语句中，表达式经过计算，并根据表达式返回的结果是真是假来

执行相应的代码。在 PHP 中，经常使用的条件语句有：if…else…else if；switch…case。下面分别对这两种条件语句进行说明。

1．IF 语句

IF 语句的基本表达式如下所示：

```
if (expr)
   statement
```

在上面的语句中，expr 按照布尔求值。如果 expr 的值为 TRUE，将执行 statement，如果值为 FALSE，则会忽略 statement。

如下示例：

```
<?php
if ($a > $b)
   print "a 大于 b";
?>
```

以上语句，判断变量 a 是否大于 b，如果 a 大于 b，则输出字符串“a 大于 b”，否则不做任何处理。

如果按照条件执行不止一条语句，当然并不需要给每条语句都加上一个 if 子句。可以将这些语句放入语句组中，通过大括号对“{”和“}”括起来。例如，如果$a 大于$b，以下代码将显示“a 大于 b”并且将$a 的值赋给$b：

```
<?php
if ($a > $b) {
   print "a 大于 b";
   $b = $a;
}
?>
```

if 语句可以无限层地嵌套在其他 if 语句中，这给程序不同部分的条件执行提供了可能。

经常需要在满足某个条件时执行一条语句，而在不满足该条件时执行其他语句，这正是 else 的功能。else 延伸了 if 语句，可以在 if 语句中的表达式的值为 FALSE 时执行语句。例如以下代码在$a 大于$b 时显示“a 大于 b”，反之则显示“a 小于 b”。

```
<?php
if ($a > $b) {
   print "a 大于 b";
} else {
   print "小于 b";
}
?>
```

如果需要同时判断多个条件，则上面的 if…else 语句则满足不了需求，PHP 提供了 elseif 的语句来扩展需求。elseif 通常用在 if 和 else 语句之间。如下面的语句，将判断变量 a 和变量 b 的大小，如果 a 大于 b，则输出“a 大于 b”；如果 a 等于 b，则输出“a 等于 b”；如果 a 小于 b，则输出“a 小于 b”。

```
<?php
if ($a > $b) {
   print "a 大于 b";
} elseif ($a == $b) {
   print "a 等于 b";
} else {
   print "a 小于 b";
```

```
}
?>
```

下面通过一个完整的实例来学习 if…elseif…else 语句的应用。

实例 2-35　IF 语句的应用

本实例介绍 IF 语句的应用。根据学生考试成绩输出不同的结果，如代码 2-35 所示。

【代码 2-35】　if.php

```
<?php
$achievement=91;

if ($achievement<60)
    echo "你不及格";
elseif ($achievement>=60 && $achievement<70)
    echo "你刚刚及格了";
elseif ($achievement>=70 && $achievement<80)
    echo "你得了良好";
elseif ($achievement>=80 && $achievement<90)
    echo "你很优秀哦!";
else
    echo "你兼职太棒了!"
?>
```

执行结果如图 2-30 所示。



图 2-30　if 语句应用实例

2．Switch 语句

Switch 语句和具有同样表达式的一系列的 IF 语句相似。很多场合下需要把同一个变量（或表达式）与很多不同的值比较，并根据它等于哪个值来执行不同的代码。这正是 Switch 语句的用途。

Switch 的语法如下：

```
switch (expr) {
  case expr1:
    statement1;
    break;
  case expr2:
    statement2;
    break;
    :
    :
  default:
    statementN;
    break;
}
```

其中的 expr 条件，通常为变量名称。而 case 后的 exprN 表示变量值。冒号后则为符合该

条件时将要执行的部分。



注意 一定要用 break 跳离循环体。

下面两个例子使用两种不同方法实现同样的结果，一个用一系列 if 语句，另一个用 switch 语句：

```
<?php
if ($i == 0) {
   print "i equals 0";
} elseif ($i == 1) {
   print "i equals 1";
} elseif ($i == 2) {
   print "i equals 2";
}

switch ($i) {
   case 0:
      print "i equals 0";
      break;
   case 1:
      print "i equals 1";
      break;
   case 2:
      print "i equals 2";
      break;
}
?>
```

为避免错误，理解 switch 是怎样执行的非常重要。switch 语句一行接一行地执行（实际上是语句接语句）。开始时没有代码被执行。仅当一个 case 语句中的值和 switch 表达式的值匹配时 PHP 才开始执行语句，直到 switch 的程序段结束或者遇到第一个 break 语句为止。如果不在 case 的语句段最后写上 break，PHP 将继续执行下一个 case 中的语句段。例如：

```
<?php
switch ($i) {
   case 0:
      print "i equals 0";
   case 1:
      print "i equals 1";
   case 2:
      print "i equals 2";
}
?>
```

这里，如果$i 等于 0，将执行所有的 print 语句；如果$i 等于 1，将执行后面两条 print 语句。只有当$i 等于 2 时，才得到“预期”的结果，即显示“i equals 2”。所以，break 语句很重要。

在 switch 语句中条件只求值一次并用来和每个 case 语句比较。在 elseif 语句中条件会再次求值。如果条件比一个简单的比较要复杂得多，或者在一个多次的循环中，那么用 switch 语句可能会快一些。

一个 case 语句也可以为空，这样只不过将控制转移到了下一个 case 语句。

```
<?php
switch ($i) {
   case 0:
   case 1:
   case 2:
      print "i is less than 3 but not negative";
      break;
   case 3:
      print "i is 3";
}
?>
```

一个 case 的特例是 default。它匹配了任何和其他 case 都不匹配的情况，并且应该是最后一条 case 语句。例如：

```
<?php
switch ($i) {
   case 0:
      print "i equals 0";
      break;
   case 1:
      print "i equals 1";
      break;
   case 2:
      print "i equals 2";
      break;
   default:
      print "i is not equal to 0, 1 or 2";
}
?>
```

实例 2-36　switch 语句的应用

本实例介绍 switch 控制语句的应用，如代码 2-36 所示。

【代码 2-36】　switch.php

```
<?php
switch (date("D")) {
  case "Mon":
    echo "今天星期一";
    break;
  case "Tue":
    echo "今天星期二";
    break;
  case "Wed":
    echo "今天星期三";
    break;
  case "Thu":
    echo "今天星期四";
    break;
  case "Fri":
```

```
        echo "今天星期五";
        break;
      default:
        echo "今天放假";
        break;
    }
    ?>
```

很明显，实例 2-36 用 if 处理将会很麻烦。在进行程序设计时，应将出现概率较大的条件放在前面，较少出现的条件放在后面，以增加程序的执行效率。上例由于每天出现的概率相同，所以不需注意条件的顺序。

代码执行结果如图 2-31 所示。



图 2-31　switch 控制语句的应用

2.5.2　循环语句

循环语句用于反复执行一系列的语句，直到条件表达式保持真值。为了保证循环的正确执行，条件表达式中计算的值应该在每次执行循环语句的时候进行修改。

下面分别介绍两种最常用的循环语句。

1．for 循环

for 循环是 PHP 中最复杂的循环结构。它的行为和 C 语言的相似。for 循环的语法是：

```
for (expr1; expr2; expr3) statement
```

第一个表达式（expr1）在循环开始前无条件求值一次。expr2 在每次循环开始前求值。如果值为 TRUE，则继续循环，执行嵌套的循环语句。如果值为 FALSE，则终止循环。expr3 在每次循环之后被求值（执行）。

每个表达式都可以为空。expr2 为空意味着将无限循环下去（和 C 一样，PHP 认为其值为 TRUE）。有时候如果想通过 break 语句来结束循环而不是用 for 的表达式真值判断，则 expr2 设置为空是个好办法。

实例 2-37　for 循环语句的应用

本实例演示 for 循环的各种常用用法（都是显示数字 1 到 10），如代码 2-37 所示。

【代码 2-37】　for.php

```
<?php
/* 应用 1，每个条件都有 */

for ($i = 1; $i <= 10; $i++) {
    print $i. "-";
}

/* 应用 2，省略第 2 个表达式 */
print "<br>";
```

```
for ($i = 1; ; $i++) {
    if ($i > 10) {
        break;
    }
    print $i. "-";
}
print "<br>";
/* 应用 3，省略 3 个表达式 */

$i = 1;
for (;;) {
    if ($i > 10) {
        break;
    }
    print $i. "-";
    $i++;
}
print "<br>";
/* 应用 4 */

for ($i = 1; $i <= 10; print $i. "-", $i++);
?>
```

其中，第一个循环看上去是比较完整的应用（或者第 4 个），另外两个 for 循环并不完整，而是使用空表达式来实现的。事实上可以用 Break 语句来控制。

代码执行结果如图 2-32 所示。

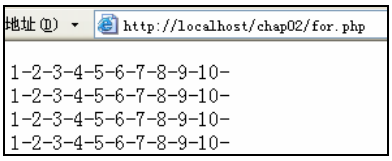


图 2-32　for 循环的应用

PHP 也支持用冒号的 for 循环的替代语法，语法格式如下：

```
for (expr1; expr2; expr3): statement; ...; endfor;
```

可以将上面的输出结果用如下语句表示：

```
/* 应用 5  */
for ($i = 1; $i <= 10; $i++) :print $i;print "-";endfor;
```

2．while 循环

while 循环是 PHP 中最简单的循环类型。它和 C 语言中的 while 类似。while 语句的基本格式是：

```
while (expr) statement
```

while 语句的含意很简单，只要 while 表达式的值为 TRUE 就重复执行嵌套中的循环语句。表达式的值在每次开始循环时检查，所以即使这个值在循环语句中改变了，语句也不会停止执行，直到本次循环结束。有时候如果 while 表达式的值一开始就是 FALSE，则循环语句一次都不会执行。

和 for 语句一样，可以在 while 循环中用花括号括起一个语句组，或者用替代语法：

```
while (expr): statement ... endwhile;
```

实例 2-38 while 循环语句的应用

本实例演示了 while 循环的应用，如代码 2-38 所示，两个应用完全一样，都显示数字 1 到 10。

【代码 2-38】 while.php

```
<?php
/* 应用 1 */
$i = 1;
while ($i <= 10) {
    print $i++ . "-";
}

/* 应用 2 */
$i = 1;
while ($i <= 10):
    print $i . "-";
    $i++;
endwhile;

/* 应用 3，通过 break 语句退出 while 循环 */
$i = 1;
while ($i<20):
    print $i . "-";
    $i++;
    if ($i>10) break;
endwhile;
?>
```

代码执行结果如图 2-33 所示。

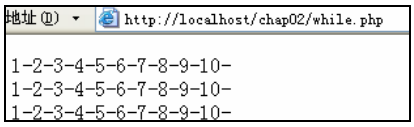


图 2-33 while 循环语句

while 循环还有另外一种使用方式 do…while。do…while 和 while 循环非常相似，区别在于表达式的值是在每次循环结束时检查而不是开始时。和正规的 while 循环主要的区别是 do…while 的循环语句保证会执行一次（表达式的真值在每次循环结束后检查），然而在正规的 while 循环中就不一定了（表达式真值在循环开始时检查，如果一开始就为 FALSE 则整个循环立即终止）。

do…while 循环只有一种语法：

```
<?php
$i = 0;
do {
   print $i;
} while ($i > 0);
?>
```

以上循环正好运行一次，因为经过第一次循环后，检查表达式的值时，其值为 FALSE（$i

不大于 0），导致循环终止。

读者可能熟悉另一种不同的 do…while 循环用法，即把语句放在 do...while(0)之中，在循环内部用 break 语句来结束循环。以下代码片段示范了此方法：

```
<?php
$i=1;
do {
   if ($i < 5) {
      print "now out put $i <br>";
   }
   $i++;
   if ($i >5) {
      break;
   }
} while(1);
?>
```

2.5.3　break 和 continue 语句

在控制语句中，break 和 Continue 是两个比较重要的语句。通过这两个语句可以增强编程的灵活性，提高编程效率，下面分别介绍这两个语句。

1．break 语句

break 结束当前 for、foreach、while、do-while 或者 switch 结构的执行。break 可以接受一个可选的数字参数来决定跳出几重循环。

下面再通过一个实例来学习 break 语句。前面介绍的都是跳出一层循环的，下面的实例是如何通过 break 语句跳出两层甚至多层循环。

实例 2-39　break 语句的应用

break 语句有一个可选参数，表示跳出几层循环。本实例重点介绍该参数的使用，如代码 2-39 所示。

【代码 2-39】　break.php

```
<?php
$i = 0;
while (++$i) {
   switch ($i) {
   case 5:
      echo "At 5<br />\n";
      break 1;  /* 只跳出 switch 循环，1 为参数 */
   case 10:
      echo "At 10; quitting<br />\n";
      break 2;  /* 跳出 switch 和 while 循环，2 为参数 */
   default:
      break;
   }
}
?>
```

代码执行结果如图 2-34 所示。

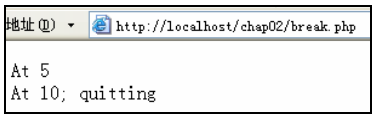


图 2-34　break 语句的应用

2．continue 语句

continue 用来跳过本次循环中剩余的代码，并在条件求值为真时开始执行下一次循环。在 PHP 中，switch 语句可以使用 continue。continue 接受一个可选的数字参数来决定跳过几重循环到循环结尾。

下面通过实例来学习 continue 语句的应用。

实例 2-40　continue 语句的应用

本实例演示 continue 语句的基本用法，同时也介绍 continue 的参数使用方法，如代码 2-40 所示。

【代码 2-40】　continue.php

```
<?php
$i=0;
while ($i++<5) {
    if ($i==2) {    // 跳出，也就是不会输出 i am 2
        continue;
    }
    echo "i am $i <br>";
}

$i = 0;
while ($i++ < 5) {
    echo "外层<br />\n";
    while (1) {
        echo "  中间层<br />\n";
        while (1) {
            echo "    内层<br />\n";
              $i=6;
            continue 3;
        }
        //因为每次到内层的时候，就跳到第一层，不会被执行
         echo "我永远不会被输出的~~.<br />\n";
    }
    echo "我也是不会被输出的 ～～.<br />\n";

}
?>
```

执行结果如图 2-35 所示。



图 2-35　continue 语句的应用

2.6　数　　组

先来考虑一个问题，如果要存储 1000 个临时数据，按照前面学习的知识，可能要定义 1000 个变量，而要定义 1000 个变量，对程序员来说，是一个很大的工作量。因此能够同时定义并保存多个数据的数组产生了。数组可以同时保存一个或多个值，比较灵活，而且处理也比较方便。

从前面已经知道，变量就是单个值的容器。但数组则是多个值的容器。一个数组也可以包括多个具有不同值的元素。PHP 还允许一个数组包括不同数据类型的元素。数组中的元素是在索引的基础上引用或操作的。数组中的索引，正常情况下都是整数。

2.6.1　数组类型

数组可以根据下标和数字元素进行分类，数组类型包括枚举数组、关联数组和多维数组。

1．枚举数组

下标为整数的数组称为枚举数组或数字索引数组。这些数组用来保存值。例如，可以有一个名为 Student 的数组，其中数组的每一个元素都保存一个学生的姓名。

在数组中，每个独立元素都可以通过索引号来引用。例如，James 这个名字可以通过索引 0 来引用。为了显示索引 2 的数组元素，可以使用下面的这个语句：

```
Echo  "student [2]
```



注意　默认情况下，数组的索引号从 0 开始，并依次增加。因此，数组最后一个索引号总是比该数组中元素总数小 1。

2．关联数组

如果需要通过名称访问数组中的元素，那么应该使用字符串作为数组的索引。可以使用一个字符串作为数组元素的索引，来保存值以及名称，使用字符串作为下标的数组称为关联数组。

要访问数组元素，可以根据其索引使用命令 Print“Student [x]”，而不是 echo（"stundent [X] "）。这里 X 表示名为 student 的数组元素的索引号。

使用数组索引（枚举）的数组和字符串索引（关联）的数组存在着微妙的区别。对于关联数组以字符串为索引，就不能计算数组中的下一个有效索引。但是，可以使用数组标识符来把索引分配给前面讨论的 student 数组。可以通过给索引赋值来创建一个关联的数组 Student_Name。

```
$Student_Name["name1"] ="James Patt";
$Student_Name["name2"] ="John Smith";
$Student_Name["name3"] ="Susan Carter";
```

也可以把现有的枚举数组转化为一个关联数组，如下所示：

```
$Student = array("name1" =>"James Patt"," name2"=>"John Smith","name3"=>"Susan Carter");
```

把枚举数组转化为关联数组之后，可以使用 echo 命令来访问数组的元素。

```
Echo  "$Student [name3]";
```

将得到输出为 Susan Carter。

还可以使用 list()和 echo()函数来访问使用字符串作为索引的数组中的元素。List()函数用

于把数组元素的值赋给变量，另一方面，echo()函数则把数组当作参数，并返回该数组每个元素的索引和值。

例如：

```
    $student = array(" name1" =>"James Patt " , " name2" =>"John Smith" , "name3"
=>"Susan Carter");
    List($key_name,$val) = echo($Student);
    Echo("$key_name");
    Echo("$val");
```

上述代码把$key_name 设置为与元素索引相等，并且$val 等于数组中的每个元素的元素值。接着使用 echo()命令显示$ key_name 和$val 的值。因此，索引将为 name1、name2 以及 name3。相应的值为 James Patt、John Smith 和 Susan Carter。

3．多维数组

可以在另外一个数组中保存不同的变量以及完整的数组，数组的元素若是数组，那么数组就是多维数组。同样，如果二维数组中的数组包含数组，便会得到一个三维数组，依此类推。虽然这些数组可以是三维数组、四维数组等，但是二维数组是最常用的。下面对多维数组进行了说明。

Roll No	1	2	3
Name	James Patt	John Smith	Susan Carter

上面说明的这个数组就是二维数组的一个例子，这里 Student 数组包含了另一个称为 Roll NO 的数组。

考虑到 Student 数组包含数组元素的情况，如果希望的访问 Student 数组中第一个元素的第二个子元素，必须使用如下的两个下标：

```
    $Sudent[0][1]
```

也可以把一个数组想象成一幅图案。例如，在一幅图案上移动鼠标光标时，需要捕获该图案的坐标。在这种情况下，需要使用多维数组来捕获 x 和 y 坐标，如下所示：

```
    Coordinates [x][y]
```

数组的维数没有限制。数组的维也可以组合。例如，可以让数组的第一维用整数作为索引，第二维使用字符串作为索引，第三维用整数作为索引等。多维数组在表示复杂的统计数据时非常有用。

实例 2-41　多维数组的应用

本实例将介绍一个简单的多维数组的应用，如代码 2-41 所示。

【代码 2-41】　multi_array.php

```
    <html>
    <body>
    <?php
    $student=array("0"=>array("name"=>"James","sex"=>"male","age"=>"28"),
    "1"=>array("name"=>"John","sex"=>"male","age"=>"25"),
    "2"=>array("name"=>"Susan","sex"=>"female","age"=>"24"));
    Print $student[2][age]
    ?>
    </body>
    </html>
```

上述代码的输出结果为 24。

2.6.2　数组初始化

数组可以用两种方法进行初始化。

1．使用数组标识符

数组标识符是一组内空的方括号。其语法如下：

```
$arr[key] = value;
$student[] = value;
// key 可以是 integer 或者 string
// value 可以为任何值
```

如果$student 不存在，将会新建一个。要改变一个值，只要给它赋一个新值。可以使用数组标识符来初始化 Student 数组，如下所示：

```
$Student [] = "James";
$Student [] = "John";
```

这里把“James”和“John”这两个值赋给了 Student 数组。注意，没有为这个数组指定下标，因此下标 0 和 1 就分别赋给了“James”和“John”。也可以显式的为数组指定下标，例如：

```
$ Student [0] = "James";
$ Student [1] = "John";
```

正常情况下，下标按顺序分配给数组。但是也可以为数组元素随机分配下标。例如：

```
$ Student [10] = "James";
$ Student [3] = "John";
```

如果要删除一个键名/值对，要对它用 unset()函数，如下代码所示：

```
<?php
$arr = array(5 => 1, 12 => 2);
$arr[] = 56;    // 等同于 $arr[13] = 56;

$arr["x"] = 42; //添加 1 个新元素 x
unset($arr[5]); // 从数组中删除该元素 5
unset($arr);    // 删除整个数组
?>
```



注意　如果给出方括号，但没有指定键名，则取当前最大整数索引值，新的键名将是该值增加 1。如果当前还没有整数索引，则键名将为 0。如果指定的键名已经有值了，该值将被覆盖。

这里所使用的最大整数键名不一定当前就在数组中。它只要在上次数组重新生成索引后曾经存在过就行了。



提示　如果给一个当前最大键名是负值的数组添加一个新值，则新生成的的索引将为 0。

2．使用 array()函数

使用 array()函数初始化数组是更简单的方法。使用 array()函数可以同时为一个数组分配多个值。可以使用 array()函数按照下面的方式来定义数组 Student：

```
$Student = array ("James", "John", "Susan" );
```



注意 在正常情况下，操作数组时使用默认下标。这意味着数组的索引正常情况下是从 0 开始，但是，也可以使用“=>”运算符重载默认下标。在上面的例子中，Student 数组有 3 个元素，其下标分别为 0、1、2。可以指定下标从 1 开始。实现这一点，可以编写下面这段代码：

$ Student = array (1=>"James", "John","Susan");

“key => value”参数对的语法如下：

```
array( [key =>]
value
    , ...
    )
// key 为键名，可以是 integer 或者 string
// value 可以是任何值
```

如下代码所示：

```
<?php
$arr = array("foo" => "bar", 12 => true);

echo $arr["foo"]; // 结果为 bar
echo $arr[12];    // 结果为 1 即 true
?>
```

key 可以是整数或者字符串。如果键名是一个整数的标准表达方法，则被解释为整数（例如"8"将被解释为 8，而"08"将被解释为"08"）。PHP 中数组下标的变量类型不会对数组造成影响，数组的类型只有一种，它可以同时包含整型和字符串型的下标。而数组的值可以是任何值。

下面的代码创建了一个二维数组：

```
<?php
$arr = array("somearray" => array(6 =>5, 13 => 9, "a" => 42));

echo $arr["somearray"][6];    // 结果为 5
echo $arr["somearray"][13];   // 结果为 9
echo $arr["somearray"]["a"];  // 结果为 42
?>
```

使用 TRUE 作为键名将使 1 成为键名。使用 FALSE 作为键名将使 0 成为键名。使用 NULL 作为键名将等同于使用空字符串。使用空字符串作为键名将新建（或覆盖）一个用空字符串作为键名的值，这和用空的方括号不一样。



提示 不能用数组和对象作为键名。这样会导致一个警告：Illegal offset type。

2.6.3 数组的应用

数组在 PHP 中应用非常广泛，接下来将介绍常用的数组编程技巧。

1．unset()函数删除数组元素

unse 函数释放给定的变量。它可以释放各种变量，包括数组的值。其声明格式如下：

```
void unset ( mixed var [, mixed var [, ...]] )
```

如下面的语句：

```
<?php
// 销毁单个变量
unset ($foo);

// 销毁单个数组元素
unset ($bar['quux']);

// 销毁一个以上的变量
unset ($foo1, $foo2, $foo3);
?>
```

在数组中介绍这个函数，主要是说明数组元素被删除后，其索引的最大值仍然被保存着。

实例 2-42　创建数组并添加删除元素

本实例重点介绍给数组赋值，然后删除数组元素，并重新给数组赋值。要注意其中数字下标索引的变化。新元素的下标是从之前曾经出现过的下标最大值开始的，无论该下标的数据是否存在。如代码 2-42 所示，文件名为 array_1.php。

【代码 2-42】　array_1.php

```
<?php
// 创建一个简单的数组
$array = array(1, 2, 3, 4, 5);
print_r($array);
echo "<br>";

// 现在删除其中的所有单元，但保持数组本身的结构
foreach ($array as $i => $value) {
    unset($array[$i]);
}
print_r($array);
echo "<br>";
// 添加一个单元（注意新的键名是 5，而不是你可能以为的 0）
$array[] = 6;
print_r($array);

// 重新索引：
$array = array_values($array);
$array[] = 7;
print_r($array);
?>
```

执行结果如图 2-36 所示。

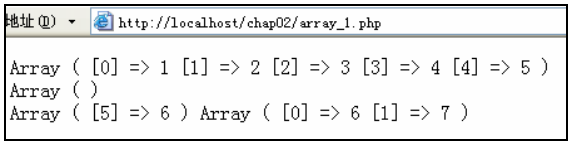


图 2-36　创建数组并添加删除元素

2．Foreach 函数

PHP 4 以上版本包括了 foreach 结构，和 Perl 以及其他语言很像。这只是一种遍历数组的

简便方法。foreach 仅能用于数组，当试图将其用于其他数据类型或者一个未初始化的变量时会产生错误。foreach 有两种语法，第 1 种比较次要但却是第 2 种的有用的扩展，如下所示：

```
foreach (array_expression as $value) statement  //第一种
foreach (array_expression as $key => $value) statement  //第二种
```

第一种格式遍历给定的 array_expression 数组。每次循环中，当前单元的值被赋给$value 并且数组内部的指针向前移一步（因此下一次循环中将会得到下一个单元）。

第二种格式做同样的事，只除了当前单元的键值也会在每次循环中被赋给变量$key。



注意 当 foreach 开始执行时，数组内部的指针会自动指向第一个单元。这意味着不需要在 foreach 循环之前调用 reset()函数。

foreach 所操作的是指定数组的一个拷贝，而不是该数组本身。因此使用 foreach()语句，原数组指针也没有变，数组单元的值也不受影响。

实例 2-43 foreach 语句的使用

本实例演示如何通过 foreach 语句输出数组中的元素。代码 2-43 中运用了 foreach 语句的两种方法。

【代码 2-43】 foreach.php

```
<?php
/* 使用第 1 中方法输出数组元素 */
$a = array (1, 2, 3, 17);

foreach ($a as $v) {
   print "\$a 的数组的当前值为：$v <br>";
}

/* 打印出数组下标*/
$b = array (1, 2, 3, 17);
$i = 0; /*初始化下标 */
foreach ($b as $v) {
    print "\$b[$i] => $v.<br>";
    $i++;
}

/* 使用第 2 种方法 */
$c = array (
    "one" => 1,
    "two" => 2,
    "three" => 3,
    "seventeen" => 17
);

foreach ($c as $k => $v) {
    print "\$c[$k] => $v<br>";
}

/* 多维数组 */
```

```
$d[0][0] = "a";
$d[0][1] = "b";
$d[1][0] = "y";
$d[1][1] = "z";

foreach ($d as $v1) {
    foreach ($v1 as $v2) {
        print "\$d 的当前值：$v2<br>";
    }
}

/*动态数组 */
foreach (array(1, 2, 3, 4, 5) as $v) {
    print "动态数组当前值:$v<br>";
}
?>
```

执行结果如图 2-37 所示。

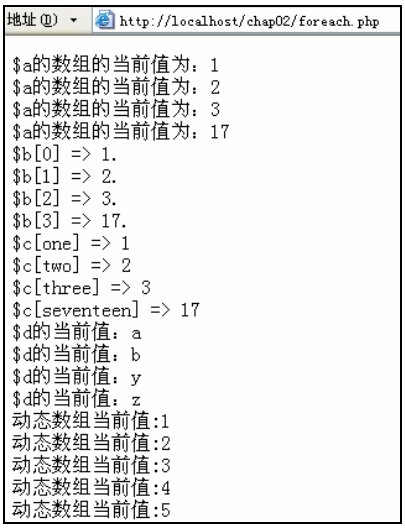


图 2-37　foreach 语句的应用

3．转换为数组

对于任何的类型：整型、浮点、字符串、布尔和资源，如果将一个值转换为数组将得到一个且仅有一个元素的数组（其下标为 0），该元素即为此标量的值。

如果将一个对象转换成一个数组，所得到的数组的元素为该对象的属性（成员变量），其键名为成员变量名。

如果将一个 NULL 值转换成数组，将得到一个空数组。

4．each 函数遍历数组内容

该函数返回数组中当前的键/值对并将数组指针向前移动一步，其声明如下所示：

```
array each ( array &array )
```

返回 array 数组中当前指针位置的键/值对并向前移动数组指针。键值对被返回为 4 个单元的数组，键名为 0、1、key 和 value。单元 0 和 key 包含有数组单元的键名，1 和 value 包含有数据。如果内部指针越过了数组的末端，则 each()返回 FALSE。

实例 2-44 each 函数语句的使用

本实例演示 each 函数的用法。首先输出 each 函数的返回内容，然后循环输出一个数组的全部元素，如代码 2-44 所示。

【代码 2-44】 each.php

```
<?php

$foo = array("bob", "fred", "jussi");
$bar = each($foo);
echo '$bar 当前的内容:';
print_r($bar);


$foo = array("Robert" => "Bob", "Seppo" => "Sepi");
$bar_1 = each($foo);
echo '<br>$bar_1 当前的内容:';
print_r($bar);

//使用 each 函数输出数组内容
$fruit = array('a' => 'apple', 'b' => 'banana', 'c' => 'cranberry');
reset($fruit);  //数组指针归 0
echo '<br>$fruit 数组全部元素:<br>';
while (list($key, $val) = each($fruit)) {  //循环输出数组内容
    echo "$key => $val<br>";
}
?>
```

代码执行结果如图 2-38 所示。

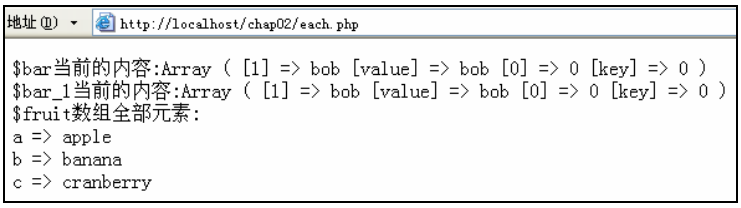


图 2-38 each 函数的应用

5．list()语句

该语句把数组中的值赋给一些变量。该函数的声明如下所示：

```
void list ( mixed varname, mixed ... )
```

list()语句用一步操作给一组变量进行赋值。其中，参数 varname 为从数组中被赋值的变量。

提示 list()仅能用于数字索引的数组并假定数字索引从 0 开始。

实例 2-45 list 语句的使用

本实例演示如何通过 list 语句访问数组的元素，如代码 2-45 所示。

【代码 2-45】　list.php

```
<?php
$info = array('coffee', 'brown', 'caffeine');

// 显示数组所有元素
list($drink, $color, $power) = $info;
echo "$drink 的颜色是 $color ，调料为 $power<br>";

//显示部分数组
list($drink, , $power) = $info;
echo "$drink 添加了调料 $power.<br>";

// 显示一个元素
list( , , $power) = $info;
echo "我需要调料：$power!<br>";
?>
```

执行结果如图 2-39 所示。

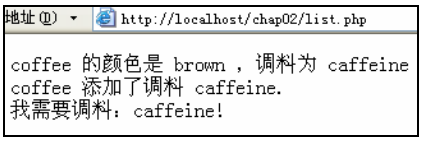


图 2-39　list 语句的应用

6．array_slice()函数取出数组的部分元素

该函数从数组中取出一段，其声明如下所示：

```
array array_slice ( array array, int offset [, int length [, bool
preserve_keys]] )
```

array_slice()返回根据 offset 和 length 参数所指定的 array 数组中的一段序列。如果 offset 非负，则序列将从 array 中的此偏移量开始。如果 offset 为负，则序列将从 array 中距离末端计算偏离。

如果给出了 length 并且为正，则序列中将具有这么多的元素。如果给出了 length 并且为负，则序列将终止在距离数组末端 length 长度的位置。如果省略，则序列将从 offset 开始一直到 array 的末端。



array_slice()默认将重置数组的键。自 PHP 5.0.2 起，可以通过将 preserve_keys 设为 TRUE 保持原来的键。

实例 2-46　使用 array_slice 函数取出部分数组元素

本实例将演示如何从一个数组中取得部分连续的元素，并保存在另外一个数组中，如代码 2-46 所示。

【代码 2-46】　array_slice.php

```
<?php
$input = array("a", "b", "two"=>"c", "d", "e");
```

```
$output = array_slice($input, 2);      // 返回 "c", "d", and "e"
$output = array_slice($input, -2, 1);  // 返回 "d"
$output = array_slice($input, 0, 3);   // 返回 "a", "b", and "c"

// 输出部分数组成员，注意数组索引同原始数组索引的变化
print_r(array_slice($input, 2, -1));
echo "<br>";
//设定 preserve_keys 参数为 true，保证数组索引和原来数组一致
//即元素 c,d 在 input 中索引为 two 和 3，则新数组索引一样为 two 和 3
print_r(array_slice($input, 2, -1, true));
?>
```

执行结果如图 2-40 所示。

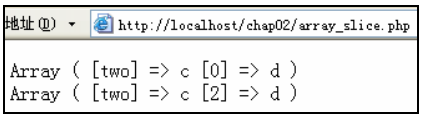


图 2-40　取出数组的部分元素

7．array_shift()函数删除数组第一个元素

该函数将数组开头的单元移出数组，其声明如下所示：

```
mixed array_shift ( array &array )
```

array_shift()将 array 的第一个元素移出并作为结果返回，将 array 的长度减一并将所有其他单元向前移动一位。所有的数字键名将改为从 0 开始计数，文字键名将不变。如果 array 为空（或者不是数组），则返回 NULL。

提示　　使用本函数后会重置 reset()数组指针。

实例 2-47　使用 array_shift 函数删除数组第一个元素

本实例演示如何将一个数组的第一个元素删除，如代码 2-47 所示。

【代码 2-47】　array_shift.php

```
<?php
$stack = array("orange", "banana", "apple", "raspberry");
$fruit = array_shift($stack);  //删除第一个元素，此时$fruit 变量值为 orange
//输出$fruit
echo "\$fruit 的值为：$fruit <br>";

//查看整个数组结构并查看数组索引变化
print_r($stack);
?>
```

执行结果如图 2-41 所示。

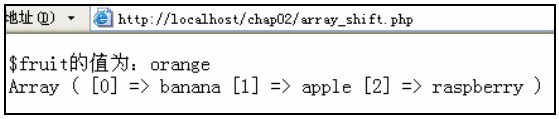


图 2-41　删除数组第一个元素

8．array_pop()函数删除数组最后一个元素

该函数将数组最后一个单元删除，声明如下所示：

```
mixed array_pop ( array &array )
```

array_pop()弹出并返回 array 数组的最后一个单元，并将数组 array 的长度减 1。如果 array 为空或者不是数组）将返回 NULL。

提示　使用本函数后会重置 reset()数组指针。

实例 2-48　使用 array_pop 函数删除数组最后一个元素

本实例演示如何删除数组最后一个元素并返回最后一个元素，如代码 2-48 所示。

【代码 2-48】　array_pop.php

```
<?php
$stack = array("orange", "banana", "apple", "raspberry");

//获取最后一个元素
$fruit = array_pop($stack);
//输出最后一个元素
echo "最后一个元素是: ". $fruit ."<br>";
//显示被删除后的数组内容
print_r($stack);
?>
```

执行结果如图 2-42 所示。

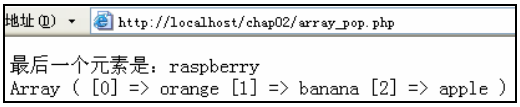


图 2-42　删除数组最后一个元素

9．array_unshift()和 array_push()函数插入元素

array_unshift 函数在数组开头插入一个或多个元素，其声明如下所示：

```
int array_unshift ( array &array, mixed var [, mixed ...] )
```

array_unshift()将传入的元素插入到 array 数组的开头。注意，元素是作为整体被插入的，因此传入元素将保持同样的顺序。所有的数值键名将修改为从 0 开始重新计数，所有的文字键名保持不变。该函数返回 array 数组新的单元数目。

array_push 函数将一个或多个单元压入数组的末尾，其声明如下所示：

```
int array_push ( array &array, mixed var [, mixed ...] )
```

array_push()将 array 当成一个栈，并将传入的变量压入 array 的末尾。array 的长度将根据入栈变量的数目增加，如下所示：

```
<?php
    $array[] = $var;
?>
```

并对每个 var 重复以上动作。该函数返回数组新的单元总数。

实例 2-49　将元素插入到数组中

本实例演示如何通过 array_unshift 和 array_push 函数将新元素插入到数组中，如代码 2-49

所示。

【代码 2-49】 add_unit.php

```
<?php
//头部插入
$a = array("1", "2");
array_unshift($a, "3", "4");
print_r($a);
echo "<br>";

//尾部插入
$b = array("banana");
array_push($b, "apple", "raspberry");
print_r($b);
?>
```

执行结果如图 2-43 所示。

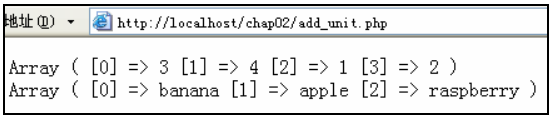


图 2-43 插入新元素

10．array_splice()函数替换元素

array_splice 函数把数组中的一部分去掉并用其他值取代，其声明如下所示：

```
array array_splice ( array &input, int offset [, int length [, array
replacement]] )
```

array_splice()把 inpu 数组中由 offset 和 length 指定的单元去掉，如果提供了 replacement 参数，则用 replacement 数组中的单元取代。返回一个包含有被移除单元的数组。注意，input 中的数字键名不被保留。

如果 offset 为正，则从 input 数组中该值指定的偏移量开始移除。如果 offset 为负，则从 input 末尾倒数该值指定的偏移量开始移除。

如果省略 length，则移除数组中从 offset 到结尾的所有部分。如果指定了 length 并且为正值，则移除这么多单元。如果指定了 length 并且为负值，则移除从 offset 到数组末尾倒数 length 为止中间所有的单元。



当给出了 replacement，要移除从 offset 到数组末尾所有单元时，用 count（$input）作为 length。

如果给出了 replacement 数组，则被移除的单元被此数组中的单元替代。如果 offset 和 length 的组合结果是不会移除任何值，则 replacement 数组中的单元将被插入到 offset 指定的位置。注意，替换数组中的键名不保留。如果用来替换的值只是一个单元，那么不需要给它加上 array()，除非该单元本身就是一个数组。

实例 2-50 使用 array_splice 函数替换数组元素

本实例演示了 array_splice()函数的一些常用方法，如代码 2-50 所示。

【代码 2-50】　array_splice.php

```
<?php
$input = array("red", "green", "blue", "yellow");
$back=array_splice($input, 2);
// $input 现在为 array("red", "green")


$input = array("red", "green", "blue", "yellow");
$back=array_splice($input, 1, -1);
// $input 现在为 array("red", "yellow")

$input = array("red", "green", "blue", "yellow");
$back=array_splice($input, 1, count($input), "orange");
// $input 现在为 array("red", "orange")

$input = array("red", "green", "blue", "yellow");
$back=array_splice($input, -3, 2, array("black", "maroon"));
//输出$back
print_r($back);
echo "<br>";
//输出$input
print_r($input);

$input = array("red", "green", "blue", "yellow");
$back=array_splice($input, 3, 0, "purple");
// $input 现在为 array("red", "green",
//         "blue", "purple", "yellow");
?>
```

执行结果如图 2-44 所示。

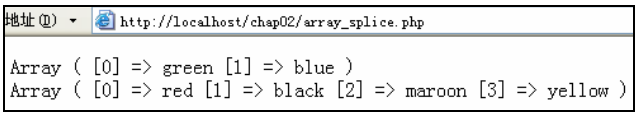


图 2-44　数组元素替换

11．浏览数组元素

下面介绍常用的浏览数组元素的函数。

（1）current()函数。current 函数返回数组中的当前元素，其声明如下：

```
mixed current ( array &array )
```

每个数组中都有一个内部的指针指向它“当前的”元素，初始指向插入到数组中的第一个单元。current()函数返回当前被内部指针指向的数组元素的值，并不移动指针。如果内部指针指向超出了单元列表的末端，current()返回 FALSE。

提示　如果数组包含有空的元素（0 或者""，空字符串）则本函数在碰到这个元素时也返回 FALSE。这使得用 current()不可能判断是否到了此数组列表的末端。要正确遍历可能含有空单元的数组，需要用 each()函数。

（2）reset()函数。rese 函数将数组的内部指针指向第一个单元，声明如下所示：

```
mixed reset ( array &array )
```

reset()将 array 的内部指针倒回到第一个元素并返回数组第一个元素的值，如果数组为空则返回 FALSE。

（3）end()函数。end 函数将数组的内部指针指向最后一个元素：

```
mixed end ( array &array )
```

end()将 array 的内部指针移动到最后一个元素并返回其值。

（4）next()函数。next 函数将数组中的内部指针向前移动一位，其声明如下所示：

```
mixed next ( array &array )
```

返回数组内部指针指向的下一个单元的值，或当没有更多单元时返回 FALSE。next()和 current()类似，只有一点区别，在返回值之前将内部指针向前移动一位。这意味着它返回的是下一个数组元素的值并将数组指针向前移动了一位。如果移动指针的结果是超出了数组单元的末端，则 next()返回 FALSE。

（5）pos()函数。该函数为 current 函数的别名。

（6）prev()函数。prev 函数将数组的内部指针倒回一位，其声明如下：

```
mixed prev ( array &array )
```

返回数组内部指针指向的前一个单元的值，或当没有更多元素时返回 FALSE。

实例 2-51　浏览数组元素

本实例演示了如何使用常用的 current、end、next、prev 以及 reset 等函数来查看数组元素或者移动数组元素，如代码 2-51 所示。

【代码 2-51】　check_array.php

```
<?php
$transport = array('foot', 'bike', 'car', 'plane');
$mode = current($transport); // $mode = 'foot';
echo "当前元素值为：" . $mode . "<br>";
$mode = next($transport);    // $mode = 'bike';
echo "当前元素值为：" . $mode . "<br>";
$mode = next($transport);    // $mode = 'car';
echo "当前元素值为：" . $mode . "<br>";
$mode = prev($transport);    // $mode = 'bike';
echo "当前元素值为：" . $mode . "<br>";
$mode = end($transport);     // $mode = 'plane';
echo "当前元素值为：" . $mode . "<br>";
reset($transport);
$mode = current($transport);  //$mode= 'foot'
echo "当前元素值为：" . $mode . "<br>";
?>
```

执行结果如图 2-45 所示。

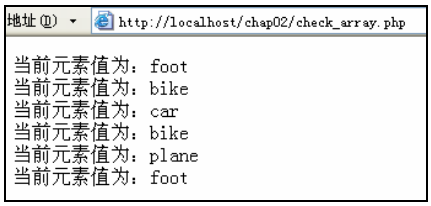


图 2-45　浏览数组元素

12．数组排序

在 PHP 中，提供了一些函数，可以很方便地对数组进行排序。

（1）sort()函数。sort 函数对数组进行排序，该函数声明如下：

```
bool sort ( array &array [, int sort_flags] )
```

当函数结束时数组元素值将被从最低到最高重新安排。



本函数为 array 中的元素赋予新的键名。这将删除原有的键名而不仅是重新排序。

如果成功则返回 TRUE，失败则返回 FALSE。

可选的第 2 个参数 sort_flags 可以用以下值改变排序的行为。

SORT_REGULAR：正常比较元素（不改变类型）。

SORT_NUMERIC：元素被作为数字来比较。

SORT_STRING：元素被作为字符串来比较。

SORT_LOCALE_STRING：根据当前的区域（locale）设置把单元当作字符串比较。

（2）asort()函数。asort 函数对数组进行排序，并保持索引关系，其声明如下：

```
bool asort ( array &array [, int sort_flags] )
```

本函数对数组进行排序，数组的索引键值保持和元素的关联。如果成功则返回 TRUE，失败则返回 FALSE。

（3）ksort()函数。ksort 函数对数组按照键名排序，其声明如下所示：

```
bool ksort ( array &array [, int sort_flags] )
```

对数组按照键名排序，保留键名到数据的关联。本函数主要用于关联数组。如果成功则返回 TRUE，失败则返回 FALSE。

（4）rsort()函数。rsort 函数对数组逆向排序，其声明如下所示：

```
bool rsort ( array &array [, int sort_flags] )
```

本函数对数组进行逆向排序（最高到最低）。如果成功则返回 TRUE，失败则返回 FALSE。

另外同正向排序一样，方向排序也提供了 arsort 函数来进行关联数组排序、krsort 函数来进行键值排序。

实例 2-52　数组排序

本实例介绍常用的数组排序函数对数组进行各种排序，如代码 2-52 所示。

【代码 2-52】　all_sort.php

```
<?php
//sort 排序
echo "sort 排序<br>";
$fruits = array("lemon", "orange", "banana", "apple");
sort($fruits);
foreach ($fruits as $key => $val) {
   echo "fruits[".$key."] = " . $val . "<br>";
}


//asort 排序
echo "asort 排序<br>";
$fruits = array("d" => "lemon", "a" => "orange", "b" => "banana", "c" =>
```

```
"apple");
    arsort($fruits);
    foreach ($fruits as $key => $val) {
        echo "$key = $val <br>";
    }

    //ksort 排序
    echo "ksort 排序<br>";
    $fruits = array("d"=>"lemon", "a"=>"orange", "b"=>"banana", "c"=>"apple");
    krsort($fruits);
    foreach ($fruits as $key => $val) {
        echo "$key = $val <br>";
    }

    //rsoft 排序
    echo "rsort 排序<br>";
    $fruits = array("lemon", "orange", "banana", "apple");
    rsort($fruits);
    foreach ($fruits as $key => $val) {
        echo "$key = $val <br>";
    }

    //arsoft 排序
    echo "arsort 排序<br>";
    $fruits = array("d" => "lemon", "a" => "orange", "b" => "banana", "c" =>
"apple");
    arsort($fruits);
    foreach ($fruits as $key => $val) {
        echo "$key = $val <br>";
    }
    //krsoft 排序
    echo "krsort 排序<br>";
    $fruits = array("d"=>"lemon", "a"=>"orange", "b"=>"banana", "c"=>"apple");
    krsort($fruits);
    foreach ($fruits as $key => $val) {
        echo "$key = $val <br>";
    }
    ?>
```

执行结果如图 2-46 所示。

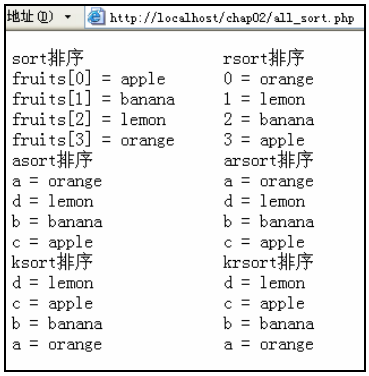


图 2-46　数组排序

13．对数组重新排序

在一些应用程序中，可能希望按照一种方式对数组排序。本节将介绍两个能够对数组重新排序的函数。

（1）shuffle()函数。shuffle 函数能够将数组打乱，进行随机排序，其声明如下所示：

```
bool shuffle ( array &array )
```

本函数为 array 中的元素赋予新的键名。这将删除原有的键名而不仅是重新排序。

（2）array_reverse()函数。array_reverse 函数返回一个元素顺序相反的数组，其声明如下所示：

```
array array_reverse ( array array [, bool preserve_keys] )
```

array_reverse()接受数组 array 作为输入并返回一个元素内容完全相同但是相反顺序的新数组，如果 preserve_keys 为 TRUE 则保留原来的键名。

实例 2-53　对数组重新排序

本实例将演示 shuffle 和 array_reverse 函数的应用，通过这两个函数能够对数组重新排序，如代码 2-53 所示。

【代码 2-53】　resort.php

```
<?php
echo "随机排序数组：<br>";
$numbers = array(1,2,3,4,5);
shuffle($numbers);
foreach ($numbers as $number) {
    echo "$number ";
}

$input  = array("php", 4.0, "hello");
$result = array_reverse($input);
//倒序排列
echo "<br>倒序排列：<br>";
print_r($result);
//保持键值倒序排列
echo "<br>保持键值倒序排列：<br>";
$result_keyed = array_reverse($input, TRUE);
print_r($result_keyed);
?>
```

执行结果如图 2-47 所示。



图 2-47　对数组重新排序

14．Range()函数自动生成数组

下面将介绍一个重要的函数，该函数能够很方便地产生设定范围的数组。

range 函数建立一个包含指定范围单元的数组，其声明如下所示：

```
array range ( mixed low, mixed high [, number step] )
```

range()返回数组中从 low 到 high 的元素，包括它们本身。如果 low 大于 high，则序列将从 high 到 low。

如果给出了 step 的值，它将被作为单元之间的步进值。step 应该为正值。如果未指定，step 默认为 1。

实例 2-54 Range 函数的应用

本实例介绍如何使用 range 函数产生数组元素，如代码 2-54 所示。

【代码 2-54】 range.php

```
<?php
// 产生 array(0, 1, 2, 3, 4, 5)
foreach (range(0, 5) as $number) {
    echo $number;
}
echo "<br>";
// array(0, 10, 20, 30, 40,)，这里设置了 step 参数
foreach (range(0, 40, 10) as $number) {
    echo $number;
}
echo "<br>";
//产生字符数组
// array('a', 'b', 'c', 'd');
foreach (range('a', 'd') as $letter) {
    echo $letter;
}
echo "<br>";
// array('c', 'b', 'a');第 1 个参数大于第 2 个参数
foreach (range('c', 'a') as $letter) {
    echo $letter;
}
?>
```

执行结果如图 2-48 所示。



图 2-48 range 函数应用

15．统计数组元素个数

下面介绍几个常用的函数来统计数组元素个数。

（1）count()函数。count 函数计算数组中的单元数目或对象中的属性个数，其声明如下所示：

```
int count ( mixed var [, int mode] )
```

（2）array_count_values()函数。array_count_values 函数统计数组中所有的值出现的次数，

其声明如下所示：

```
array array_count_values ( array input )
```

array_count_values()返回一个数组，该数组用 input 数组中的值作为键名，以在 input 数组中出现的次数作为值。

实例 2-55　统计数组元素个数

本实例使用两个函数来统计数组元素的个数，如代码 2-55 所示。

【代码 2-55】　count.php

```
<?php
$a =array(1,3,5);
$result = count($a);
// $result == 3
echo "\$a 的元素个数为：" . $result;
$b[0] = 7;
$b[5] = 9;
$b[10] = 11;
$result = count($b);
// $result = = 3;
echo "<br>\$b 的元素个数为：" . $result . "<br>";

$result = count(null);
// $result = = 0

$result = count(false);
// $result = = 1

//array_count_values 函数，统计每个元素出现的次数
$array = array(1, "hello", 1, "world", "hello");
print_r(array_count_values ($array));
?>
```

执行结果如图 2-49 所示。

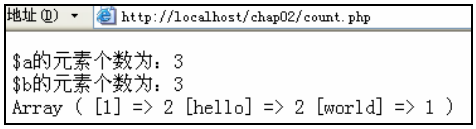


图 2-49　统计数组元素个数

从图中可以看出，hello 出现两次，1 出现两次。

2.7　函　　数

函数是自包含的和独立的代码块。它可以完成某种具体的任务。一般情况下，这个任务需要多次的执行。由于函数是独立的代码块，可以独立的运行，因此它没有必要属于给定脚本的一部分。

函数在执行的时候一般遵循以下过程：

（1）从函数的脚本接受一个参数；

（2）根据函数代码进行处理并得到预定的结果；

（3）返回结果，并把控制返回到调用它的脚本。

PHP 的函数有以下一些特性。

（1）无需在引用函数之前定义它。即可以在脚本中很早就调用一个函数，但是等到很晚的时候才定义这个函数。

（2）如果已经声明了一个函数，不能再次定义或者取消定义。这是因为 PHP 不支持函数重载。因此必须使用惟一的名称定义每个函数。

（3）PHP 不允许函数支持参数个数可变的情况。

（4）PHP 函数支持默认参数。

在 PHP 中，大致有 3 种函数类型，即自定义函数、内置函数以及变量函数。

2.7.1 内置函数

PHP 有很多标准的函数和结构。还有一些函数需要和特定的 PHP 扩展模块一起编译，否则在使用它们的时候就会得到一个致命的“未定义函数”错误。例如，要使用诸如 imagecreatetruecolor()的“图像函数”，需要在编译 PHP 的时候加上 GD 的支持。或者，要使用 mysql_connect()函数，就需要在编译 PHP 的时候加上 MySQL 支持。另外还有一些核心函数，例如“字符串函数”和“变量函数”，它们存在于每一个版本的 PHP 中。调用 phpinfo()或者 get_loaded_extensions()可以得知 PHP 加载了哪些扩展库。同时还应该注意，很多扩展库默认就是有效的。

下面将介绍一些 PHP 中使用频率比较高的函数和语法结构。

1．Echo 函数

该函数用于输出一个或者多个字符串，其使用格式如下：

```
void echo ( string arg1 [, string argn...])
```

严格来说 echo 并不是一个函数，而是一个语法结构，因此，在调用该函数的时候，后面可以不使用括号将参数扩起来。

实例 2-56 echo 语句的应用

本实例介绍 echo 语句的各种应用，如代码 2-56 所示。

【代码 2-56】 echo.php

```
<?php
echo "Hello World<br>";  //直接输出字符串

//输出多行文字
echo "This spans
multiple lines.
well done.<br>";

//输出引号
echo "Escaping characters is done \"Like this\".<br>";


//直接在字符串中增加变量
```

```
$foo = "foobar";
$bar = "barbaz";

echo "foo is $foo<br>"; // 输出 foo is foobar

// 输出数组
$car = array("value" => "foo");

echo "this is {$car['value']} !<br>"; // 输出 this is foo !

// 如果用单引号括出一个变量，则只会输出变量名称，而不是变量的值
echo 'foo is $foo<br>'; // foo is $foo

// 直接输出变量
echo $foo ."<br>";         // 输出 foobar
echo $foo,$bar;     // 输出 foobarbarbaz
echo "<br>";
// 使用多参数输出，也就是将每个单词分开作为参数
echo 'This ', 'string ', 'was ', 'made ';
echo 'This ' . 'string ' . 'was ' . 'made ' . "<br>";



// echo 不是一个函数，因此下面的语法是错误的
//($bar) ? echo 'true' : echo 'false';

//而 print 是一个函数，下面的语法正确
($bar) ? print('true'): print('false');

$myname="forway";
$yourname="admin";
?>
<br>html 中应用变量：<br>
我的名字是：<?php echo $myname ?>
<br>
你的名字是：<?php  echo $youname ?>
```

执行结果如图 2-50 所示。

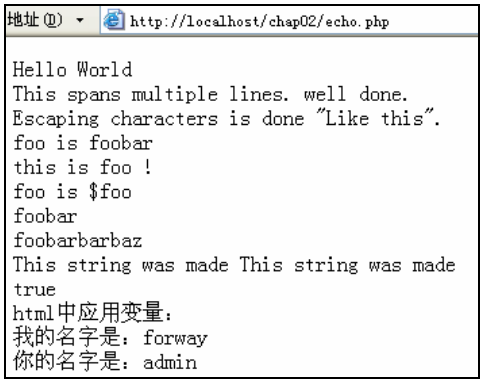


图 2-50　echo 函数的应用

提示 还有一个重要的函数就是 exit 函数，该函数类似 echo，只是使用该函数输出内容之后，会终止后面代码的执行，在很多时候，该函数非常有用。exit 函数还有一个别名就是 die 函数。

2．print 函数

另外一个常用的输出函数名为 print，其声明格式如下所示：

```
int print ( string arg )
```

该函数总是返回 1。严格来说 print 也不是一个函数，而是一个语法结构，因此输出的时候，参数不需要括号。

实例 2-57 print 语句的应用

本实例演示 print 语句的应用，其用法同 echo 类似，如代码 2-57 所示。

【代码 2-57】 print.php

```
<?php
//直接输出字符串
print("Hello World");
print "<br>";
//可以分成多行写字符串
print "i
want to
fly high";
print "<br>";

//输出引号
print "这一句输出引号 \"我在引号中\".";

// 字符串中包含变量
$foo = "foobar";
$bar = "barbaz";
print "<br>";

print "foo is $foo"; // foo is foobar
print "<br>";
//输出一段文字
print <<<END
This uses the "here document" syntax to output
multiple lines with $variable interpolation. Note
that the here document terminator must appear on a
line with just a semicolon no extra whitespace!
END;
?>
```

3．include 函数

include 函数在 PHP 网页设计中非常重要。它可以很好地实现代码的可重用性，同时还可以简化文件代码。

include 语句包含并运行指定文件。假如现在有文件 A.php，在 A.php 文件中，包含了 B.php 文件，也就是在 A.php 文件中加入了“include 'B.php';”语句。当服务器执行 A.php 到这一行

的时候，就会自动读取 B.php 文件中的代码并执行其中的代码。

此方法非常有用，如在网页设计中，很多时候，大部分页头和页尾都是一样的，为了减少每个页面的代码重复量，可以将页尾和页头分别做成 header.htm 和 footer.htm，然后在页面中包含该页面，如下所示：

```
<?php
include 'header.htm';
//其他代码
include 'footer.php';
?>
```

实例 2-58　简单的 include 应用

本实例将介绍 include 函数的最基本应用，其中包含了两个文件，一个文件名为 a_inc.php，另一个文件为 b_inc.php。a_inc.php 如代码 2-58 所示，该文件声明了几个变量。

【代码 2-58】　a_inc.php

```
<?php
$who = '张三';
$fruit = '苹果';
?>
```

b_inc.php 文件包含了 a_inc.php 文件，并调用了 a_inc.php 文件中的变量，如代码 2-59 所示。

【代码 2-59】　b_inc.php

```
<?php

echo "$who 喜欢吃 $fruit"; // 输出喜欢吃，此时没有包含
echo "<br>";
include 'a_inc.php';

echo "$who 喜欢吃 $fruit";// 输出张三喜欢吃苹果
?>
```

执行 b_inc.php，结果如图 2-51 所示。



图 2-51　简单的包含文件执行结果

从图中可以看出，没有包含文件之前，变量值为空，而包含后，从 a_inc.php 文件中读出变量值并显示。

如果 include 出现于调用文件中的一个函数里，则被调用的文件中所包含的所有代码将表现得如同它们是在该函数内部定义的一样。所以它将遵循该函数的变量范围。

实例 2-59　函数内部包含文件

本实例演示在一个函数内部包含另外一个文件，该实例由两个文件组成，文件名称分别

为 a_inc.php 和 c_inc.php。a_inc.php 文件用于声明变量，如代码 2-58 所示。

c_inc.php 用于声明一个函数，并包含 a_inc.php 文件，如代码 2-60 所示。

【代码 2-60】 c_inc.php

```
<?php
function foo()
{
    global $fruit;  //声明一个全局变量

    include 'a_inc.php';

    echo "$who 喜欢吃 $fruit";
}

//调用函数
foo();                    // 输出张三喜欢吃苹果
echo "<br>";
//函数外面调用 a_inc.php 文件中的变量
//因为$fruit 为全局变量，所以可以访问
//而$who 只在函数内部能访问，在外部不能访问
//只能输出喜欢吃苹果
echo "$who 喜欢吃 $fruit";

?>
```

c_inc.php 文件的执行结果如图 2-52 所示。



图 2-52 函数中包含其他文件

因为 include()和 require()是特殊的语言结构，在条件语句中使用必须将其放在语句组中（花括号中）。如下代码所示：

```
<?php

//以下用法错误
if ($condition)
    include $file;
else
    include $other;
// 以下用法正确
if ($condition) {
    include $file;
} else {
    include $other;
}
?>
```

可以在被包括的文件中使用 return()语句来终止该文件中程序的执行，并返回调用它的脚本。同样也可以从被包含的文件中返回值。可以像普通函数一样获得 include 调用的返回值。

因为 include()是一个特殊的语言结构，其参数不需要括号。在比较其返回值时要注意。如下面代码所示：

```
<?php
// 以下代码有问题，不应该跟括号
if (include(a_inc.php') == 'OK') {
    echo 'OK';
}

// 工作正常
if ((include b_inc.php') == 'OK') {
    echo 'OK';
}
?>
```

实例 2-60　Return 的应用

本实例将介绍如何在被包含文件使用 Return 语句来返回值以及终止被包含文件的执行，该实例由两个文件组成，文件名分别为 d_inc.php 和 e_inc.php。

d_inc.php 文件如代码 2-61 所示，该文件使用 return 关键字返回一个值。

【代码 2-61】　d_inc.php

```
<?php
$a=5;
if ($a==5)
    echo '我是老五<br>';
else
    echo '我不是老五<br>';

$b='我被返回啦!';
return $b;

echo "我不会被执行啦!";
?>
```

e_inc.php 文件如代码 2-62 所示，该文件包含 d_inc.php 文件，并输出 d_inc.php 的返回值。

【代码 2-62】　e_inc.php

```
<?php
echo '接下来要包含的 d_inc.php 文件<br>';
$back=include 'd_inc.php';
echo $back;
?>
```

执行 e_inc.php 文件，结果如图 2-53 所示。



图 2-53　return 关键字的使用

从输出结果可以看出，被包含文件使用 return 后，不会执行后面的代码。另外包含文件

能够获得从被包含文件的返回值。如果被包含文件没有返回值，则$back 可能为 1，表示调用成功。如果失败则为 0。



如果不小心在文件中两次包含了同一个文件，则 PHP 编译器会提出警告信息。为了避免发生这种情况，可以使用 include_once()函数。其同 include()语句类似，惟一区别是如果该文件中的代码已经被包含了，则不会再次包含。如同此语句名字暗示的那样，只会包含一次。

4．require()函数

require()函数和 include()函数除了处理失败之外在各方面都完全一样。include()产生一个警告，而 require()则导致一个致命错误。换句话说，如果想在丢失文件时停止处理页面，应该使用 require()。而 include()不会这样，脚本会继续运行。

代码 2-63 所示由于包含了 a.php 不存在，所以终止执行，并显示错误信息。

【代码 2-63】 require.php

```
<?php
require 'a.php';
echo "我不会被执行!";
?>
```

由于 a.php 文件不存在，所以会产生错误消息，如图 2-54 所示。

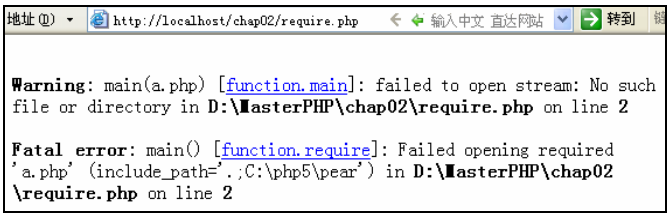


图 2-54 require 错误信息

从图中可以看出，由于 a.php 不存在，所以代码停止执行，而“我不会被执行!”没有被输出。另外该语句也会产生错误提示“Fatal error”。

代码 2-64 是通过 include 函数包含了 a.php，但是它只显示警告信息，仍然继续执行代码。

【代码 2-64】 require_1.php

```
<?php
include 'a.php';
echo "我会被执行!";
?>
```

执行结果如图 2-55 所示。

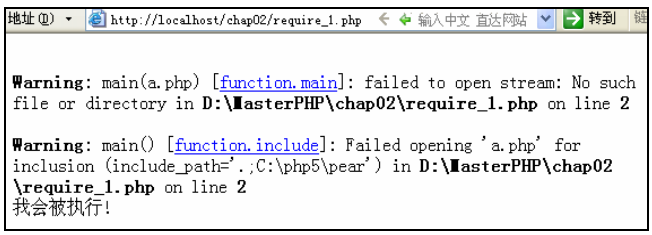


图 2-55 include 的警告信息

从图中可以看出，虽然 a.php 不存在，但是系统只是显示警告信息，并且继续执行代码，输出了“我会被执行语句!”。

有关 PHP 内置函数，就介绍到这里，其他的函数，会在相关的章节专门介绍。读者也可以参考 PHP 的参考手册。

2.7.2　自定义函数

用户自定义函数也称自定义函数，它们不是 PHP 提供的，是由程序员创建的。由于自己创建了这样的函数，所以就可以完全控制这些函数。因此可以让一个函数完全按照自己希望的方式运行。

1．声明函数

在 PHP 中，定义函数的方法同其他编程语言几乎一样。下面是 PHP 声明函数的语法结构：

```
Function function_name($argument1,$ argument2, $argument3,…, $argumentn)
{
    //函数代码 code
    Return  返回值;
}
```

以上语法结构中，关键字的含义如下。

（1）function：用于声明用户自定义函数的关键字。

（2）function_name：要创建的函数名称。该名称将在以后被调用时使用。函数名应该惟一，因为 PHP 不支持重载。在命名函数的时候，需要遵循和变量命名相同的原则。但是函数名不能以$开头，而变量可以。

（3）argument：要传递给函数的值。函数可以有多个参数，它们之间用逗号分开。但是参数项是可选的，可以在调用函数的时候不传递任何参数。

（4）code：是在函数被调用的时候执行的一段代码。如果有两条或者多条语句，则代码必须用大括号“{}”括起来。但是，如果只有一条代码，则不需要大括号。

（5）Return：将调用的代码需要的值返回。任何类型都可以返回，包括列表和对象。这导致函数立即结束它的运行，并且将控制权传递回它被调用的行。

2．无参数函数

下面通过简单的实例来学习无参数函数的使用。

实例 2-61　无参数函数应用

本实例演示了如何创建一个函数并调用该函数，该函数没有返回值，同时也没有参数。如代码 2-65 所示。

【代码 2-65】　func_1.php

```
<?php
function foo ()
{
   echo "天气好热<br>";
    echo "希望明天下雨<br>";
}
Echo "函数还没有被调用呢!<br>";
Foo();  //开始调用该函数
```

```
Echo "函数调用成功!";
?>
```

以上的函数很简单，没有返回值，也没有传递参数。执行结果如图 2-56 所示。



图 2-56　简单函数应用

3．有参数函数

PHP 支持按值传递参数（默认），通过引用传递和默认参数值。可变长度参数列表仅在 PHP4 和后续版本中被支持。

（1）值传递参数。按值进行参数传递是 PHP 的默认传递方式。使用这个方法，必须在主程序调用时传递一个值（参数）。

实例 2-62　值传递参数

本实例演示如何通过值来传递参数，调用函数，实现了将两个数相加，如代码 2-66 所示。

【代码 2-66】　sum.php

```
<?php
function sum ($a,$b)
{
   Echo $a+$b;
}
sum(10,20);  //开始调用该函数
?>
```

运行该文件，输出结果如图 2-57 所示。



图 2-57　值参数传递

按值传递，比较容易理解，即直接将数据传递到函数。

（2）引用传递。在按照值传递的时候，只有参数的副本传递给被调用的函数。但是在被调用函数内部对这些值的任何修改，都不会影响调用函数中的原始值。引用传递其实也就是地址传递，将一个变量的地址作为参数传递。

实例 2-63　修改变量的值

本实例演示了在函数内部修改一个参数的值，而实际并没有修改到该参数变量真正的结果，如代码 2-67 所示。

【代码 2-67】　valuechange.php

```
<?php
$myNum=100;
function Valuechange ($number)
```

```
{
    $number=$number+1;
     Echo $number . "<br>";
}
Valuechange($myNum);
Echo $myNum;
?>
```

执行结果如图 2-58 所示。



图 2-58　修改参数值

从上面的结果可以看出，虽然在函数内部将变量$number 的值增加了，但是实际上并没有改变$myNum 的值。

很多时候，将一个参数传递给函数，而且该参数的新值要返回给调用程序，由于值传递不能实现，可以通过引用传递来进行。

通过引用传递参数比较简单，只需要直接在参数前面添加“&”符号就可以。

实例 2-64　引用传递参数

本实例演示了通过引用传递参数，这样会将传递的参数的原始值修改，如代码 2-68 所示。

【代码 2-68】　addresschange.php

```
<?php
$myNum=100;
function addresschange($number)
{
    $number=$number+1;
     Echo $number. "<br>";
}
addresschange(&$myNum);
Echo $myNum;
?>
```

输出结果如图 2-59 所示。



图 2-59　引用传递参数

从输出结果可以看到，通过引用传递，变量$myNum 的值在函数内部被改变了。

（3）默认值传递。使用默认的参数值传递方法，函数必须在调用的时候有一个参数。如果没有要使用的值，就把默认值传递给函数参数。

默认值必须是常量表达式，不是变量、类成员或者函数调用。

提示　当使用默认参数时，任何默认参数必须放在非默认参数的右侧；否则，可能函数不会按照预期的情况工作。

实例 2-65　函数的默认值

本实例介绍如何使用函数的默认值，如代码 2-69 所示。

【代码 2-69】　func_default.php

```
<?php
//默认值的应用
function hobby($style = "运动")
{
    return "我喜欢 $style<br>";
}
echo hobby ();
echo hobby ("唱歌");


function hobby_1 ($who, $type = "跳舞")
{
    return "$who 喜欢 $type<br>";
}

echo hobby_1("张三");
echo hobby_1("张三","逛街");

// 默认值的不正确使用，默认值必须在其他参数的右边
//因此会产生警告信息
function hobby_2($type = "跳舞",$who)
{
    return "$who 喜欢 $type<br>";
}

echo hobby_2("张三");
//但是下面的调用方法可以
//但是默认值就失去了意义
//echo hobby_2("爬山","张三");
?>
```

执行结果如图 2-60 所示。

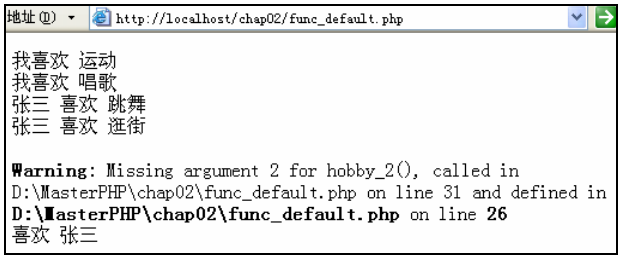


图 2-60　默认参数的应用

从图中可以看出，前面两个函数正常运行，而最后一个函数，则出现参数警告，显示结果为“喜欢张三”。因为此时默认参数没有发挥作用。但是如果改成“hobby_2("爬山","张三")”

还是可以运行的，但默认参数就失去了意义。

4．函数返回值

如果在一个函数中调用 return()语句，将立即结束此函数的执行，并将它的参数作为函数的值返回。return()会终止 eval()语句或者脚本文件的执行。

如果在全局范围中调用，则当前脚本文件中止运行。如果当前脚本文件是被 include()或者 require()，则控制交回调用文件。此外，如果当前脚本是被 include()，则 return()的值会被当作 include()调用的返回值。如果在主脚本文件中调用 return()，则脚本中止运行。如果当前脚本文件是在 php.ini 中的配置选项 auto_prepend_file 或者 auto_append_file 所指定的，则此脚本文件中止运行。


注意　既然 return()是语言结构而不是函数，并不需要用括号将参数括起来。事实上不用括号比用括号更常见，尽管用哪一种并无所谓。

实例 2-66　函数的返回值

本实例介绍函数返回值的应用，如代码 2-70 所示。

【代码 2-70】　func_back.php

```
<?php
//1 个返回值
function square ($num)
{
   return $num * $num;
}
echo "返回值为:".square(4) . '<br>';


//通过数组返回多个数值
function small_numbers()
{
   return array (0, 1, 2);
}
list ($zero, $one, $two) = small_numbers();
echo "返回值为：<br>$zero <br> $one <br> $two";
?>
```

执行结果如图 2-61 所示。



图 2-61　函数的返回值

2.7.3　变量函数

PHP 支持变量函数的概念，这意味着如果一个变量名后有圆括号，PHP 将寻找与变量的

值相同的函数，并且将尝试执行它。这个可以被用于实现回调函数和函数表等。

变量函数不能用于语言结构，例如 echo()、print()、unset()、isset()、empty()、include()、require()以及类似的语句。需要使用自己的外壳函数来将这些结构用作变量函数。

实例 2-67　变量函数的应用

本实例演示如何创建一个变量函数并应用，如代码 2-71 所示。

【代码 2-71】　func_var.php

```
<?php
function foo()
{
    echo "foo()函数<br>";
}

function bar($arg = '')
{
    echo "bar()函数，参数为 $arg <br>";
}

// 创建一个输出函数
function echoit($string)
{
    echo $string;
}

$func = 'foo';
$func();       // 将调用 foo()

$func = 'bar';
$func('hello');  //调用 bar()

$func = 'echoit';
$func('test');  // 调用 echoit 函数
?>
```

执行结果如图 2-62 所示。



图 2-62　变量函数

2.7.4　函数变量的作用域

为了让读者理解作用域的概念，下面通过两段代码来说明。

```
<?php
$a = 1; /* 全局参数 */
```

```
function Test()
{
    echo $a; /* 本地参数 */
}

Test();
?>
```

以上脚本不会有任何输出，因为 echo 语句引用了一个局部版本的变量$a，而且在这个范围内，它并没有被赋值。读者可能注意到 PHP 的全局变量和 C 语言有一点不同，在 C 语言中，全局变量在函数中自动生效，除非被局部变量覆盖。这可能引起一些问题，有些人可能漫不经心的改变一个全局变量。PHP 中全局变量在函数中使用时必须声明为全局。

如下面的代码：

```
<?php
function Test()
{
    $a=110;
}

Echo $a;
?>
```

同理，以上代码也不会有任何输出。因为$a 是在一个函数中被赋值的，其作用域只有在函数的内部，当在函数外部访问该变量的时候，是无法访问的。

虽然变量的作用域限制了变量的访问能力。但是还是可以从函数外部，访问函数内部的变量，从而使得脚本更加灵活以及动态性能更好。PHP 允许使用两个关键词 global 和 static 来实现这一点。

1．Global 语句

Global 语句允许从脚本的任何位置访问函数内部定义的变量以及从函数内部访问函数意外的变量。因此在声明变量时，必须在变量前面加上 global 关键字。

实例 2-68　使用全局变量

本实例演示如何使用一个已经声明的全局变量，如代码 2-72 所示。

【代码 2-72】　global_1.php

```
<?php
$a = 1;
$b = 2;

function Sum()
{
    global $a, $b;  //必须使用 global 关键字声明

    $b = $a + $b;
}

Sum();
```

```
echo $b;  //输出结果 3
?>
```

以上脚本的输出将是 3。在函数中声明了全局变量$a 和$b，任何变量的所有引用变量都会指向到全局变量。对于一个函数能够声明的全局变量的最大个数，PHP 没有限制。

在全局范围内访问变量的第二个方法，是用特殊的 PHP 自定义$GLOBALS 数组。

实例 2-69　使用$GLOBALS 数组

本实例演示如何通过$GLOBALS 全局变量数组来获取全局变量值，如代码 2-73 所示。

【代码 2-73】　global_2.php

```
<?php
$a = 1;
$b = 2;

function Sum()
{
    $GLOBALS["b"] = $GLOBALS["a"] + $GLOBALS["b"];
}

Sum();
echo $b;  //返回 3
?>
```

2．Static 语句（静态变量）

变量范围的另一个重要特性是静态变量（static variable）。静态变量仅在局部函数域中存在，当程序执行离开此作用域时，其值并不丢失。

看看下面的例子：

```
<?php
function Test ()
{
    $a = 0;
    echo $a;
    $a++;
}
?>
```

本函数没什么用处，因为每次调用时都会将$a 的值设为 0 并输出“0”。将变量加 1 的$a++没有作用，因为一旦退出本函数则变量$a 就不存在了。

要写一个不会丢失本次计数值的计数函数，要将变量$a 定义为静态的：

```
<?php
function Test()
{
    static $a = 0;
    echo $a;
    $a++;
}
?>
```

这样，每次调用 Test()函数都会输出$a 的值并加 1。

2.8　日期和时间处理

日期和时间的处理是 PHP 编程不可缺少的一部分。很多时候都需要对时间进行编程，如显示当前的时间、将时间保存进数据库、从数据库中根据时间进行查询等。

2.8.1　获取日期和时间

在 PHP 中，要获取当前的日期和时间，非常简单，只需要使用 Date()函数。

Date()函数的声明如下所示：

```
string date( string format [, int timestamp] )
```

返回按照指定格式显示的时间字符串。其中参数 format 为显示格式，而参数为 timestamp 时间戳。如果没有给出时间戳则使用本地当前时间。换句话说，timestamp 是可选的，默认值为 time()。

提示　要将字符串表达的时间转换成时间戳，应该使用 strtotime()。

其中 format 的选项很多，具体设置如表 2-9 所示。

表 2-9　format 参数的设置

参　数　值	说　　明	应　　用
日期格式设置		
d	月份中的第几天，有前导零的 2 位数字	01 到 31
D	星期中的第几天，文本表示，3 个字母	Mon 到 Sun
j	月份中的第几天，没有前导零	1 到 31
l（“L”的小写字母）	星期几，完整的文本格式	Sunday 到 Saturday
N	ISO-8601 格式数字表示的星期中的第几天（PHP 5.1.0 新加）	1（表示星期一）到 7（表示星期天）
S	每月天数后面的英文后缀，2 个字符	st，nd，rd 或者 th。可以和 j 一起用
w	星期中的第几天，数字表示	0（表示星期天）到 6（表示星期六）
z	年份中的第几天	0 到 366
星期		
W	ISO-8601 格式年份中的第几周，每周从星期一开始（PHP 4.1.0 新加的）	例如：42（当年的第 42 周）
月		
F	月份，完整的文本格式，例如 January 或者 March	January 到 December
m	数字表示的月份，有前导零	01 到 12
M	3 个字母缩写表示的月份	Jan 到 Dec
n	数字表示的月份，没有前导零	1 到 12
t	给定月份所应有的天数	28 到 31
年		
L	是否为闰年	如果是闰年为 1，否则为 0
o	ISO-8601 格式年份数字。这和 Y 的值相同，只除了如果 ISO 的星期数（W）属于前一年或下一年，则用那一年。（PHP 5.1.0 新加）	Examples: 1999 or 2003
Y	4 位数字完整表示的年份	例如：1999 或 2003

续表

参 数 值	说 明	应 用
y	2 位数字表示的年份	例如：99 或 03
时间		
a	小写的上午和下午值	am 或 pm
A	大写的上午和下午值	AM 或 PM
B	Swatch Internet 标准时	000 到 999
g	小时，12 小时格式，没有前导零	1 到 12
G	小时，24 小时格式，没有前导零	0 到 23
h	小时，12 小时格式，有前导零	01 到 12
H	小时，24 小时格式，有前导零	00 到 23
i	有前导零的分钟数	00 到 59>
s	秒数，有前导零	00 到 59>
时区		
e	时区标识（PHP 5.1.0 新加）	例如：UTC，GMT，Atlantic/Azores
I	是否为夏令时	如果是夏令时为 1，否则为 0
O	与格林威治时间相差的小时数	例如：+0200
T	本机所在的时区	例如：EST，MDT
Z	时差偏移量的秒数。UTC 西边的时区偏移量总是负的，UTC 东边的时区偏移量总是正的。	–43200 到 43200
完整的日期 / 时间		
c	ISO 8601 格式的日期（PHP 5 新加）	2004-02-12T15:19:21+00:00
r	RFC 822 格式的日期	例如：Thu, 21 Dec 2000 16:01:07 +0200
U	从 Unix 纪元（January 1 1970 00:00:00 GMT）开始至今的秒数	参见 time()

实例 2-70 date 函数的应用

本实例将演示如何通过 date 函数来输入各种形式的日期，如代码 2-74 所示。

【代码 2-74】 date.php

```
<?php
// 输出类似：Monday
echo date("l") . "<br>";

// 输出类似：Monday 15th of August 2005 03:12:46 PM
echo date('l dS \of F Y h:i:s A'). "<br>";
echo date("F j, Y, g:i a"). "<br>";
echo date("m.d.y"). "<br>";
echo date("j, n, Y"). "<br>";
echo date("Ymd"). "<br>";
echo date('h-i-s, j-m-y, it is w Day z '). "<br>";
echo date('\i\t \i\s \t\h\e jS \d\a\y.'). "<br>";
echo date("D M j G:i:s T Y"). "<br>";
echo date('H:m:s \m \i\s\ \m\o\n\t\h'). "<br>";
echo date("H:i:s"). "<br>";
?>
```

输出结果如图 2-63 所示。

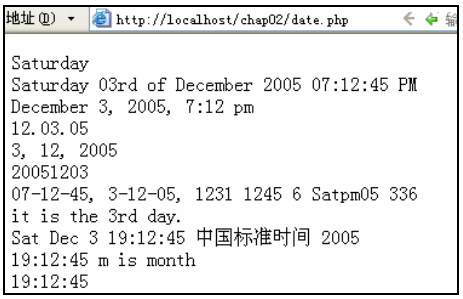


图 2-63　date 函数的应用

2.8.2　使用 getdate 函数获得日期信息

getdate 函数用于取得日期（时间）信息，其声明如下所示：

```
array getdate ( [int timestamp] )
```

返回一个根据 timestamp 得出的包含有日期信息的结合数组。如果没有给出时间戳则认为是当前本地时间。返回的数组元素如表 2-10 所示。

表 2-10　返回数组的成员

键　名	说　明	应　用
"seconds"	秒的数字表示	0 到 59
"minutes"	分钟的数字表示	0 到 59
"hours"	小时的数字表示	0 到 23
"mday"	月份中第几天的数字表示	1 到 31
"wday"	星期中第几天的数字表示	0（表示星期天）到 6（表示星期六）
"mon"	月份的数字表示	1 到 12
"year"	4 位数字表示的完整年份	例如：1999 或 2003
"yday"	一年中第几天的数字表示	0 到 365
"weekday"	星期几的完整文本表示	Sunday 到 Saturday
"month"	月份的完整文本表示	January 到 December
0	自从 Unix 纪元开始至今的秒数，和 time()的返回值以及用于 date()的值类似	系统相关，典型值为从–2147483648 到 2147483647

如下面的代码：

```
<?php
$today = getdate();
print_r($today);
?>
```

输出的结果如下所示：

```
Array (
[seconds] => 37
[minutes] => 28
[hours] => 19
[mday] => 3
[wday] => 6
[mon] => 12
[year] => 2005
[yday] => 336
[weekday] => Saturday
```

```
    [month] => December
    [0] => 1133609317 )
```

2.8.3 使用 mktime 函数取得一个日期的时间戳

mktime 函数用于取得一个日期的 Unix 时间戳，其声明如下：

```
    int mktime ( [int hour [, int minute [, int second [, int month [, int day
[, int year [, int is_dst]]]]]]] )
```

根据给出的参数返回 Unix 时间戳。时间戳是一个长整数，包含了从 Unix 纪元（January 1 1970 00:00:00 GMT）到给定时间的秒数。

参数可以从右向左省略，任何省略的参数会被设置成本地日期和时间的当前值。

参数的含义如下。

hour：小时数。

minute：分钟数。

second：秒数（1min 之内）。

month：月份数。

day：天数。

year：年份数，可以是两位或四位数字，0～69 对应于 2000～2069，70～100 对应于 1970～2000。在如今系统中普遍把 time_t 作为一个 32 位有符号整数的情况下，year 的合法范围是 1901 到 2038 之间，不过此限制自 PHP 5.1.0 起已被克服了。

is_dst：本参数可以设为 1，表示正处于夏时制时间（DST），0 表示不是夏时制，或者–1（默认值）表示不知道是否是夏时制。如果未知，PHP 会尝试自己搞明白。这可能产生不可预知（但并非不正确）的结果。如果 PHP 运行的系统中启用了 DST 或者 is_dst 设为 1，某些时间是无效的。例如 DST 自 2:00 生效，则所有处于 2:00 到 3:00 之间的时间都无效，mktime() 会返回一个未定义（通常为负）的值。某些系统（例如 Solaris 8）的 DST 在午夜生效，则 DST 生效当天的 0:30 会被计算为前一天的 23:30。

下面通过两个小实例来介绍 mktime()函数的应用。

例如下面例子中每一行都会产生字符串“Jan-01-1998”。

```
    <?php
    echo date("M-d-Y", mktime(0, 0, 0, 12, 32, 1997));
    echo date("M-d-Y", mktime(0, 0, 0, 13, 1, 1997));
    echo date("M-d-Y", mktime(0, 0, 0, 1, 1, 1998));
    echo date("M-d-Y", mktime(0, 0, 0, 1, 1, 98));
    ?>
```

任何给定月份的最后一天都可以被表示为下个月的第 0 天，而不是–1 天。下面两个例子都会产生字符串“The last day in Feb 2000 is: 29”。

```
    <?php
    $lastday = mktime(0, 0, 0, 3, 0, 2000);
    echo strftime("Last day in Feb 2000 is: %d", $lastday);
    $lastday = mktime(0, 0, 0, 4, -31, 2000);
    echo strftime("Last day in Feb 2000 is: %d", $lastday);
    ?>
```

第 3 章　字符串和正则表达式

前面已经介绍了字符串的概念以及如何进行字符串的基本赋值以及字符串的输入输出技术。字符串处理在 PHP 编程中占有着重要的角色，因此本章将专门介绍如何进行各种复杂的字符串处理。

3.1　格式化字符串

为了让字符串在输出的时候，达到用户期望的显示结果，需要将字符串进行格式化处理。下面分别介绍常用的字符串格式化技术。

3.1.1　去除多余空格

用户在输入数据的时候，经常会无意中输入空格，或者有些情况下，在字符串中不允许出现空格。这个时候需要去除字符串的空格，在 PHP 中提供了以下函数去除字符串中的空格。

（1）trim()函数。该函数可以去除字符串开始位置以及结束位置的空格，并返回去掉空格后的字符串。该函数声明如下：

```
string trim ( string str [, string charlist])
```

默认的情况下，该函数去除的字符如下。

" " (ASCII 32 码为(0x20))：空格。

"\t" (ASCII 码为 9 (0x09))：Tab 字符。

"\n" (ASCII 码为 10 (0x0A))：换行符。

"\r" (ASCII 码为 13 (0x0D))：回车符。

"\0" (ASCII 码为 0 (0x00))：空字节。

"\x0B" (ASCII 为 11 (0x0B))：垂直制表符。

如果想通过该函数过滤掉特殊的字符，可以指定第二个参数。

实例 3-1　trim 函数的应用

本实例演示 trim 函数的应用，如代码 3-1 所示。

【代码 3-1】　trim.php

```
<?php
//只过滤默认字符，"点"不会被过滤
$text = "\t\t 我爱 php :) ...  ";

echo trim($text);          // 输出"These are a few words :) ..."
echo "<BR>";
//过滤掉/t 和.
```

```
    echo trim($text, " \t.");   // 输出"These are a few words :)"

    // 可以使用下面语句取出二进制数据的前后控制字符
    // (从 0 到 31)
    //$clean = trim($binary, "\x00..\x1F");

    ?>
```

执行结果如图 3-1 所示。



图 3-1　trim 函数应用

（2）ltrim()函数。ltrim 函数用于去除字符串左边的空格或者指定字符串。其默认去除的字符同 trim 一样。该函数声明如下：

```
    string ltrim ( string str [, string charlist])
```

实例 3-2　ltrim 函数的应用

本实例演示如何运用 ltrim 函数去除字符串左边设定的字符，如代码 3-2 所示，文件名为 ltrim.php。

【代码 3-2】　ltrim.php

```
    <?php
    //删除左边默认字符
    $text = "  ...我喜欢 php :) ...";
    $trimmed = ltrim($text);
    echo $trimmed;
    echo "<br>";
    //要删除.和空格
    $trimmed = ltrim($text, ". ");
    echo $trimmed;
    ?>
```

执行结果如图 3-2 所示。



图 3-2　ltrim 函数应用

（3）rtim()函数。rtrim 函数同 ltrim 函数刚好相反，该函数去掉字符串右边的字符。用法同 trim 和 ltrim 函数，其声明如下：

```
    string rtrim ( string str [, string charlist])
```

3.1.2　格式化字符串显示

下面介绍如何格式化字符串的显示。

1．字符串大小写转换函数

（1）strtolower()函数。该函数将传入的字符串参数所有的字符都转换成小写，并以小写

形式放回这个字符串。该函数声明如下：

```
string strtolower(string str)
```

实例 3-3　将字符转换成小写

本实例演示如何通过 strtolower 函数将字符转换成小写，如代码 3-3 所示。

【代码 3-3】　strtolower.php

```
<?php
$str = "I want To FLY";
$str = strtolower($str);
echo $str;
?>
```

执行结果如下：

```
i want to fly
```

（2）strtoupper()函数。该函数作用同 strtolower 函数刚好相反，是将传入的字符串参数的字符全部转换成大写，并以大写的形式返回这个字符串。该函数声明如下：

```
string strtoupper ( string string)
```

实例 3-4　将字符转换成大写

本实例演示了如何通过 strtoupper 函数将字符串中的全部字符转换成大写，如代码 3-4 所示。

【代码 3-4】　strtoupper.php

```
<?php
$str = "i want to Fly";
$str = strtoupper($str);
echo $str;
?>
```

执行结果如下：

```
I WANT TO FLY
```

（3）ucfirst()函数。该函数的作用是将字符串的第一个字符改成大写，该函数返回首字符大写的字符串。其声明如下：

```
string ucfirst ( string str)
```

实例 3-5　将字符串首字符转换成大写

该函数演示了如何通过 ucfirst 函数将字符串首字符转换成大写，如代码 3-5 所示。

【代码 3-5】　ucfirst.php

```
<?php
//原来首字符小写
$foo = 'hello world!';
$foo = ucfirst($foo);       //输出 Hello world!
echo $foo . "<br>";
//首字符大写，则不改变
$bar = 'HELLO WORLD!';
```

```
$bar = ucfirst($bar);    //输出 HELLO WORLD!
echo $bar . "<br>";
//将所有字符变成小写后，再将首字符编程大写
$bar = ucfirst(strtolower($bar)); //输出 Hello world!
echo $bar . "<br>";
?>
```

执行结果如图 3-3 所示。



图 3-3　将首字符转换成大写

（4）ucwords()函数。该函数将传入的字符串中的每个单词的首字符变成大写。如“hello world”，经过该函数处理后，将返回“Hello World”。该函数声明如下：

```
string ucwords ( string str)
```

实例 3-6　将字符串中每个单词的首字符转换成大写

本实例介绍如何通过 ucwords 函数将字符串中每个单词的首字符转换成大写，如代码 3-6 所示。

【代码 3-6】　ucwords.php

```
<?php
$foo = 'hello world!';
$foo = ucwords($foo);  //输出 Hello World!
echo $foo . "<BR>";

$bar = 'HELLO WORLD!';
$bar = ucwords($bar);           // 输出 HELLO WORLD!
echo $bar . "<BR>";
//全部转换成小写再将首字符转换成大写
$bar = ucwords(strtolower($bar)); // 输出 Hello World!
echo $bar . "<BR>";
?>
```

执行结果如图 3-4 所示。



图 3-4　ucwords 函数应用

2．HTML 文本格式化

接下来介绍几个处理 HTML 格式的函数，使用这些函数，能够有效地控制 HTML 文件的输出。

（1）nl2br()函数。该函数将传入的字符串参数中的换行符替换成 XHTML 标准中的“
”。因为在浏览器中输出字符串的时候，浏览器不能区别换行符，因此如果直接输出有好几行文本的字符串，在浏览器中显示的时候只有一行。

该函数声明如下：

```
string nl2br ( string string)
```

实例 3-7　nl2br 函数的应用

本实例演示如何通过 nl2br 函数将字符串中的换行符转换成 HTML 中的换行标记
，如代码 3-7 所示。

【代码 3-7】　nl2br.php

```
<?php
//一行输出
echo ("foo isn't\n bar");
echo "<br>";
//将\n 替换成<br />则两行输出
echo nl2br("foo isn't\n bar");
?>
```

输出结果如下：

```
foo isn't bar
foo isn't
bar
```

（2）htmlspecialchars()函数。该函数将一些特殊的字符转化成 HTML 文本实体，即字符不进行 HTML 解释，而是被当作普通文本输出。该函数的声明如下：

```
string htmlspecialchars ( string string [, int quote_style [, string charset]])
```

函数的第一个参数是要进行处理的字符串，第二个可选参数 quote_style 表示的是进行转换的方式，通常该函数主要对单引号和双引号来进行转换。quote_style 的取值如表 3-1 所示。

该函数对删除潜在的、不安全的用户数据是非常有用的。它把特殊的 HTML 字符转换成 HTML 实体，并以普通文本显示出来，用于避免输入到网站中的恶意的脚本。

提示　htmlspecialchars 函数的缺点是，某些标注在用户创建的内容中有时候是允许的。如果应用程序允许某些但并非所有的标注，则使用 strip_tags()函数更为方便。

表 3-1　quote_style 参数的取值

quote_style 参数	说　明	quote_style 参数	说　明
ENT_COMPAT	只有双引号被转换	ENT_NOQUOTES	任何一种引号都不被转换
ENT_QUOTES	单引号和双引号都被转换		

被转换后的结果如下。

“&”被转换成“&”。

“"”被转换成“'"”，如果第二个参数不是设置成 ENT_NOQUOTES。

“'”被转换成“'”，如果第二个参数被设置成 ENT_QUOTES。

“<”被转换成“<”。

“>”被转换成“>”。

htmlspecialchars 函数的第 3 个参数定义在转换过程中使用的字符集。默认的字符集是 ISO-8859-1。

PHP 支持的字符集如表 3-2 所示。

表 3-2 PHP 支持的字符集

字 符 集	别 名	说 明
ISO-8859-1	ISO8859-1	西欧，Latin-1
ISO-8859-15	ISO8859-15	西欧，Latin-9.
UTF-8		ASCII 字符集
cp866	ibm866, 866	DOS 字符集
cp1251	Windows-1251, win-1251, 1251	Windows-specific Cyrillic 字符集
cp1252	Windows-1252, 1252	西欧
KOI8-R	koi8-ru, koi8r	俄罗斯
BIG5	950	中国台湾
GB2312	936	中国简体
BIG5-HKSCS		中国香港
Shift_JIS	SJIS, 932	日本
EUC-JP	EUCJP	日本

实例 3-8 把特殊的 HTML 字符转换成普通文本

本实例使用 htmlspecialchars 函数将一些 HTML 标记转换成文本实体，如代码 3-8 所示。

【代码 3-8】 htmlspecialchars.php

```
<?php
$new = htmlspecialchars("<a href='test'>Test</a>", ENT_QUOTES);
echo $new; //文本实体：&lt;a href=&#039;test&#039;&gt;Test&lt;/a&gt;
?>
```

执行结果如图 3-5 所示。

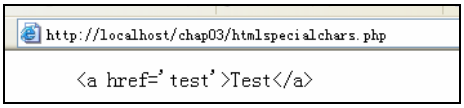


图 3-5 显示 HTML 标记

用户可以右键单击该页面，选择菜单【查看源文件】，可以看到源文件内容如下：

```
&lt;a href=&#039;test&#039;&gt;Test&lt;/a&gt;
```

提示 如果要将全部的可以转换的字符转换成文本实体，可以使用 htmlentities 函数。

htmlentities 函数结构与 htmlspecialchars 函数类似，下面通过实例来学习。

实例 3-9 htmlentities 函数的应用

本实例演示通过 htmlentities 函数将所有文本实体转换成 HTML 实体，如代码 3-9 所示。

【代码 3-9】 htmlentities.php

```
<?php
$str = "A 'quote' is <b>bold</b>";
// 输出: A 'quote' is &lt;b&gt;bold&lt;/b&gt;
echo htmlentities($str);
```

```
// 输出: A &#039;quote&#039; is &lt;b&gt;bold&lt;/b&gt;
echo htmlentities($str, ENT_QUOTES);
?>
```

通过浏览器的输出结果，可以发现两句话的显示效果一样，但是查看源代码，会发现是不一样的，其中第 1 句，单引号没有被转成文本实体，而第 2 句被转了，如下所示：

```
A 'quote' is &lt;b&gt;bold&lt;/b&gt;
A &#039;quote&#039; is &lt;b&gt;bold&lt;/b&gt;
```

（3）strip_tags()函数。strip_tags 函数将传入的字符串参数中的所有的 PHP 和 HTML 标记去掉。函数声明如下：

```
string strip_tags ( string str [, string allowable_tags])
```

如果想保留某些特定的 PHP 或者 HTML 标记，可以在第二个参数 allowable_tags 中指定。

实例 3-10　去掉所有的 HTML 和 PHP 标记

本实例演示如何使用 strip_tags 函数去掉所有的 HTML 和 PHP 标记，如代码 3-10 所示。

【代码 3-10】　strip_tags.php

```
<?php
$text = '
<p>Test paragraph.</p>
<!-- Comment -->
Other text';

echo strip_tags($text);

echo "<BR>-------<BR>";

// 允许 <p>保留
echo strip_tags($text, '<p>');
?>
```

输出结果如图 3-6 所示。

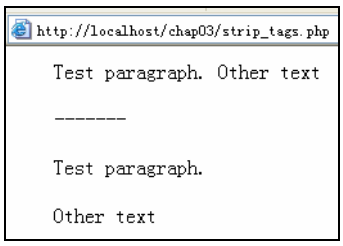


图 3-6　strip_tags 函数应用

从图中可以看出，第一次输出没有换行，是因为<p>标记被去掉了，而第 2 次输出换行了，是因为函数中设置保留了<p>标记。

源代码如下所示：

```
Test paragraph.

Other text<BR><BR>-------<BR>
```

```
<p>Test paragraph.</p>

Other text
```

3．格式化字符串以方便数据库存储

在后面章节中会学习到数据库编程。数据库编程是网络编程中的最重要的内容。将字符串保存进数据库的时候，需要对字符串中特殊的字符进行处理，否则有可能被数据库当成是控制符号。

通常，有问题的字符是单引号、双引号、反斜杠（\）和空字符（NULL）。

为了不让数据库引起误解，同时又能够让数据库保存这些字符，必须用一定的转换方法。默认的做法就是在这些控制字符的前面添加一个反斜杠，例如双引号“"”转义成“\"”，单引号“'”转义成“\'”，反斜杠“\”转义成“\\”，而如果字符串中包含两个反斜杠“\\”，则该原则仍然适用，在保存进数据库的时候，被改成“\\\”。

在 PHP 中，提供了两个函数来进行这种处理，而不需要开发人员来手动处理。

（1）addslashes()函数。该函数能够按照上面的原则对字符串参数进行转义。该函数声明如下：

```
string addslashes ( string str)
```

实例 3-11　转义数据库中的控制字符

本实例演示如何通过 addslashes 函数将保存在数据库中的数据库控制字符进行转义，如代码 3-11 所示。

【代码 3-11】　addslashes.php

```
<?php
$str = "你名叫 '张三嘛?";
echo addslashes($str);
?>
```

输出结果如下：

```
你名叫 \'张三嘛?
```

（2）stripslashes()函数。该函数同 addslashes 函数刚好相反，它去除字符串中的反斜杠。该函数声明如下：

```
string stripslashes ( string str)
```

stripslashes 函数返回去除反斜杠后的字符串。当从数据库中读取数据的时候，可以使用该函数，使得恢复本来的数据。

实例 3-12　去掉被转义的字符串中的斜杠

本实例演示如何去掉被转义字符串中的斜杠，代码 3-12 所示。

【代码 3-12】　stripslashes.php

```
<?php
$str = "你的名字叫 \'张三?";
echo stripslashes($str);
?>
```

执行结果如下：

```
你的名字叫 '张三?
```

3.2　字符串的连接和分割

很多时候，需要将一个包含有多个信息模块的字符串的信息分离开来，比如一个字符串中包含有联系人的姓名、地址、电话以及电子邮件等。在 PHP 中，也提供了若干进行字符串连接和分割的函数。

（1）explode()函数。该函数的功能是按照指定的分隔符将一个字符串分开。该函数声明如下：

```
array explode ( string separator, string string [, int limit])
```

explode 函数一共有 3 个参数，其中第一个参数是分隔符号，也就是按照什么原则来进行分隔，如果分隔符被设置为空，则函数返回 FALSE，如果要被分隔的字符串中不包含分隔符，则返回整个原始字符串；第二个参数表示要被分隔的字符串；第三个参数用来限制被分隔后的字符串片断的数量。被分隔后的字符串被存储在一个数组中。

实例 3-13　分隔字符串

本实例演示如何通过 explode 函数将字符串按照指定的分隔符进行分隔，如代码 3-13 所示。

【代码 3-13】　explode.php

```
<?php
echo "通过空格分隔字符串：<br>";
$pizza  = "a1 a2 a3 a4 a5 a6";
//通过空格分隔
$pieces = explode(" ", $pizza);
//输出返回数组的头两个元素
echo $pieces[0] . "<BR>"; // 输出 a1
echo $pieces[1]. "<BR>"; // 输出 a2

// 将分隔后的元素保存到 list 的变量中
echo "将变量保存在 list 变量中:<br>";
$data = "foo:*:1023:1000::/home/foo:/bin/sh";
list($user, $pass, $uid, $gid, $gecos, $home, $shell) = explode(":", $data);
echo $user. "<BR>"; // 输出 foo
echo $pass. "<BR>"; // 输出*

echo "限制分隔的字符串数:<BR>";
$limit="a;b;c;d;e;f;";
$back=explode(";",$limit,3);
print_r($back);
?>
```

执行结果如图 3-7 所示。

（2）implode()函数。该函数作用刚好和 explode 函数的作用相反，将一些字符串通过指定的分隔符连成一个字符串。该函数声明如下：

```
string implode (string glue, array pieces)
```

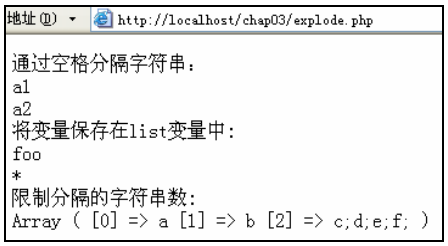


图 3-7 分隔字符串

该函数有两个参数，第一个参数 glue 表示连接字符串的连接符号，第二个参数表示需要连接成字符串的数组。

实例 3-14 连接字符串

本实例使用 implode 函数将数组元素连接成字符串，如代码 3-14 所示。

【代码 3-14】 implode.php

```
<?php
$myarray = array('lastname', 'email', 'phone');
//用逗号分隔符连接数组
$comma_separated = implode(",", $myarray);
echo $comma_separated;
?>
```

输出结果如下所示：

```
lastname,email,phone
```

提示 implode 函数还有一个别名函数 join，这两个函数用法和含义一样。

（3）strtok()函数。该函数与前面介绍的 explode 函数有些类似，但是该函数一次只从字符串中去除一些字符段。对于一次从字符串中取出一个单词的需求，该函数比 explode 函数更为有效和简单。函数声明如下：

```
string strtok ( string arg1, string arg2)
```

其中参数 arg1 是需要被分隔的字符串，而第二个参数 arg2 为分隔符。需要注意的是，被分隔的字符串会根据分隔字符串中的每个字符来进行分隔，而不是根据整个分隔字符串来分隔。在进行分隔的时候，只有第一次分隔需要第一个参数，即被分隔字符串。第一次分隔后，系统会自动记录下来第一次分隔后的指针位置，如果继续调用该函数，则会从新的指针位置来进行分隔，因此第二次调用该函数，可以省略第一个参数，只需要第二个参数即可。如果想重置指针为起始状态，则可以重新将原始字符串作为参数传给该函数。

实例 3-15 一次从字符串中取出一个单词

本实例介绍如何通过 strtok 函数一次从一个字符串中取出一个单词，如代码 3-15 所示。

【代码 3-15】 strtok.php

```
<?php
$string = "This is\tan example\nstring";
```

```
/* 使用 tab 和换行符号作为分隔符 */
$tok = strtok($string, " \n\t");
while ($tok) {
    echo "Word=$tok<br />";
    $tok = strtok(" \n\t");
}
?>
```

执行结果如图 3-8 所示。

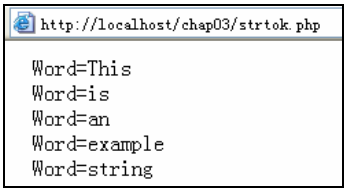


图 3-8　strtok 函数应用

（4）substr()函数。该函数允许访问指定的一个字符串的起始位置和结束位置的子字符串。比如，要获取第 3 个到第 8 个字符之间的字符串，可以采用该函数。该函数声明如下：

```
string substr ( string string, int start [, int length])
```

该函数有 3 个参数，其中第一个参数是原始字符串，第二个参数是子字符串的起始位置，第三个参数为子字符串的长度。该函数返回需要获取的子字符串。如果省略第二个参数，即起始位置，则默认从 0 开始，即从字符串的第一个字符开始，如果省略第三个参数，则默认获取从起始位置之后的所有字符。

实例 3-16　获取指定子字符串

本实例使用 substr 函数获取指定的子字符串，如代码 3-16 所示。

【代码 3-16】　substr.php

```
<?php
$rest = substr("abcdef", 1);    // 返回 "bcdef"
$rest = substr("abcdef", 1, 3); //返回"bcd"
$rest = substr("abcdef", 0, 4); //返回"abcd"
$rest = substr("abcdef", 0, 8); //返回"abcdef"

// 字符串也可以直接通过索引直接访问其字符
$string = 'abcdef';
echo $string{0};              //返回 a
echo $string{3};              //返回 d

//使用负数作为起始位置
$rest = substr("abcdef", -1);    //返回"f"
$rest = substr("abcdef", -2);    //返回"ef"
$rest = substr("abcdef", -3, 1); //返回"d"

//使用负数作为字符串长度
$rest = substr("abcdef", 0, -1);  //返回"abcde"
$rest = substr("abcdef", 2, -1);  //返回"cde"
$rest = substr("abcdef", 4, -4);  //返回""
```

```
$rest = substr("abcdef", -3, -1); //返回"de"
?>
```

提示 在后面正则表达式中，还将介绍另外一个字符串分隔函数 split()。

3.3 字符串的比较

直接比较字符串是否完全一致，可以使用“==”来进行，但是有时候可能需要进行更加复杂的字符串比较，如部分匹配等。

（1）strcmp()函数。该函数进行字符串之间的比较。函数声明如下：

```
int strcmp ( string str1, string str2)
```

该函数对传入的两个字符串参数进行比较，如果两个字符串完全相同，则返回 0；如果按照字典顺序 str1 在 str2 后面，则返回一个正数；如果 str1 小于 str2，则返回一个负数。

实例 3-17 比较字符串的大小

本实例使用 strcmp 函数来比较字符串的大小，如代码 3-17 所示。

【代码 3-17】 strcmp.php

```
<?php
$a="i want to fly";
$b="i want to climb";
$back=strcmp($a,$b);
if ($back>0)
    echo '$a 大于$b';
elseif ($back<0)
    echo '$a 小于$b';
else
    echo '$a 等于$b';
?>
```

执行结果如下：

```
$a 大于$b
```

（2）strcasecmp()函数。该函数同 strcmp 函数基本一致，但是该函数在比较的时候，不区分大小写，而 strcmp 在比较的时候是区分大小写的。该函数声明如下：

```
int strcasecmp ( string str1, string str2)
```

实例 3-18 不区分大小写比较字符串

本实例使用 strcasecmp 函数对两个字符串进行忽略大小写的比较，如代码 3-18 所示。

【代码 3-18】 strcasecmp.php

```
<?php
$var1 = "Hello";
$var2 = "hello";
if (strcasecmp($var1, $var2) == 0) {
   echo '$var1 和 $var2 相同（忽略字符串的大小写）';
```

```
}
?>
```

输出结果如下：

```
$var1 和 $var2 相同（忽略字符串的大小写）
```

（3）strnatcmp ()函数。该函数同 strcmp 函数用法基本一致，但是比较的原则有所不同。该函数并不是按照字典顺序排列的，而是按照“自然排序”比较字符串。所谓自然排序就是按照人们的习惯来进行排序。例如用 strcmp 函数来进行排序，“4”会大于“14”，而在现实中，数字“14”要大于“4”，因此 strnatcmp 函数是按照后者来进行比较的。

该函数声明如下：

```
int strnatcmp ( string str1, string str2)
```

实例 3-19　按照自然排序进行字符串比较

本实例利用 strnatcmp 函数来按照自然排序原则对字符串进行比较，如代码 3-19 所示。

【代码 3-19】　strnatcmp.php

```
<?php
$arr1 = $arr2 = array("img12.png", "img10.png", "img2.png", "img1.png");
echo "正常比较:<br>";
//usort 函数表示按照指定的函数进行排序
//此处表示用 strcmp 函数对数组元素进行排序
usort($arr1, "strcmp");
print_r($arr1);
echo "<br>按照自然数比较<br>";
//使用 strnatcmp 函数对数组进行排序
usort($arr2, "strnatcmp");
print_r($arr2);
?>
```

执行结果如图 3-9 所示。

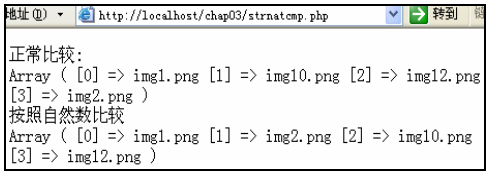


图 3-9　自然数进行字符串比较

（4）strnatcasecmp()函数。该函数同 strcasecmp 函数用法一致，只是该函数不区分大小写。函数的声明如下：

```
int strnatcasecmp ( string str1, string str2)
```

3.4　字符串的查找和替换

大部分的文本编辑器如 Word、记事本等，都会提供字符串的查找和搜索功能。本节将会介绍如何在 PHP 中实现字符串的查找和替换。

3.4.1　查找字符串

PHP 中提供了多个函数以方便在字符串中查找子字符串。

（1）strstr()函数。该函数用于在一个字符串中查找匹配的字符串或者字符。该函数声明如下：

```
string strstr ( string haystack, string needle)
```

其中第一个参数 haystack 表示是原始字符串，第二个参数 needle 表示要被查询的关键字，即子字符串或者字符。如果找到了一个 needle 的一个匹配，则该函数返回从 needle 开始到整个字符串结束的子字符串，如果没有匹配，则返回 FALSE。如果存在不止一个匹配，则返回从第一个匹配位置之后的所有子字符串。

实例 3-20　字符串查找

本实例演示了如何通过 strstr 函数进行字符串的查找，如代码 3-20 所示。

【代码 3-20】　strstr.php

```
<?php
$email = 'zhangsan@hotmail.com';
$domain = strstr($email, '@');
echo $domain;
?>
```

输出结果如下：

```
@hotmail.com
```

提示　strstr 函数还有另外一个别名函数即 strchr 函数，这两个函数用法和含义一样。

（2）stristr()函数。该函数同 strstr 函数用法基本一致，只是该函数是不区分大小写的。函数声明如下：

```
string stristr ( string haystack, string needle)
```

实例 3-21　不区分大小写进行字符串查找

本实例利用 stristr 函数进行不区分大小写的查找，如代码 3-21 所示。

【代码 3-21】　stristr.php

```
<?php
$email = 'zhangsan@hotmail.com';
$domain = stristr($email, 'S');
echo $domain;
?>
```

执行结果如下：

```
san@hotmail.com
```

（3）strrchr()函数。该函数同 strstr 函数的用法基本一样，只是从最后一个被搜索到的字符串开始返回。函数声明如下：

```
string strrchr (string haystack, char needle)
```

实例 3-22　查找的最后一个字符串

本实例演示了如何通过 strrchr 函数返回最后一个被搜索到的子字符串，如代码 3-22 所示。

【代码 3-22】　strrchr.php

```
<?php
// 获取系统路径的最后部分
echo "系统设置路径为：".$PATH;  //全部设定路径
echo "<br>";
$dir = strrchr($PATH, ":");
//获取最后路径
echo "<br>最后一个路径为:<br>".$dir;
echo "<br><br>";
//获取最后一行
echo "最后一行为：<br>";
$text = "Line 1\nLine 2\nLine 3";
$last = strrchr($text, 10);
echo  $last;
?>
```

执行结果如图 3-10 所示。

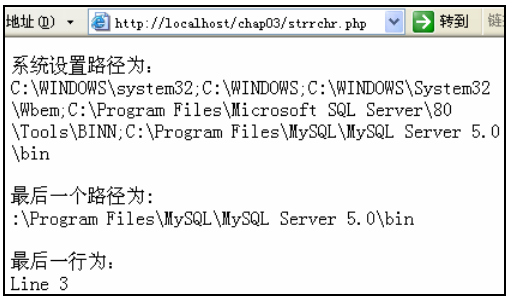


图 3-10　返回最后一个被查询的子字符串

3.4.2　定位字符串

在上一节介绍的字符串函数中，函数的执行结果都是返回字符串。下面介绍的函数，虽然也是查找字符串，但是函数返回的是子字符串在原字符串中的位置。

（1）strpos()函数。该函数在原始字符串中查找目标子字符串第一次出现的位置。该函数声明如下：

```
int strpos ( string haystack, string needle [, int offset])
```

该函数返回第一次出现参数 needle 的位置。如果没有找到字符串 needle，则返回 FALSE。其中参数 offset 表示从原始字符串 haystack 的第 offset 个字符开始搜索。

实例 3-23　定位子字符串

本实例通过 strpos 函数获取被查询字符串在原始字符串中的位置，如代码 3-23 所示。

【代码 3-23】　strpos.php

```
<?php
$mystring = 'abc';
$findme   = 'a';
$pos = strpos($mystring, $findme);
```

```
// 注意判断返回值，要用恒等表达式===
//因为如果查找到为第 1 个字符，其位置索引为 0，和 false 是一样的
if ($pos === false) {
    echo "没有找到字符串 $findme";
} else {
    echo "找到子字符串$findme";
    echo " 其位置为 $pos<br>";
}

// 设定起始搜索位置
$newstring = 'abcdef abcdef';
$pos = strpos($newstring, 'a', 1); // $pos = 7
echo "设定初始查询位置：";
echo $pos;
?>
```

执行结果如图 3-11 所示。



图 3-11　查找子字符串的位置

（2）strrpos()函数。该函数同 strpos 函数用法基本一致。只是返回最后一次出现被查询子字符串的位置。函数声明如下：

```
int strrpos ( string haystack, string needle [, int offset])
```

实例 3-24　返回最后一个被查询子字符串的位置

本函数演示了如何通过 strrpos 函数返回被搜索子字符串在原始字符串中的最后出现位置，如代码 3-24 所示。

【代码 3-24】　strrpos.php

```
<?php
$mystring="adsfdgq4ertadbasdbbasdb";
$pos = strrpos($mystring, "b");
if ($pos === false)
    echo "没有找到字符 b";
else
     echo "b 最后出现的位置为 $pos";
?>
```

输出结果如下：

```
b 最后出现的位置为 22
```



如果被查询的子字符串不在原始字符串中，strpos 和 strrpos 函数都会返回 FALSE。因为在 PHP 中 FALSE 等价于 0，也就是说字符串的第一个字符。为了避免这个问题，可以采用“= = =”来测试返回值，判断返回值是否为 FALSE，即“if ($result= = =FALSE)”。

（3）strripos 函数。该函数返回最后一次出现被查询子字符串的位置，该函数是大小写不

敏感的，即不区分大小写。该函数声明如下：

```
int strripos(string haystack, string needle [, int offset])
```

其应用与 strrpos 函数基本一样，这里省略实例。



注意　以上的几个函数中的 offset 参数不能为负数，否则会停止搜索。

（4）stripos()函数。该函数用法与 strripos 基本一致，也是不区分大小写，但是返回的是第一次出现被查询子字符串的位置。该函数声明如下：

```
int stripos ( string haystack, string needle [, int offset])
```

省略应用实例。

3.4.3　字符串替换

本小节介绍如何将目标子字符串替换成用户指定的字符串内容。

（1）str_replace()函数。该函数将用新的子字符串替换原始字符串中被指定要替换的字符串。该函数的声明如下：

```
mixed str_replace ( mixed search, mixed replace, mixed subject [, int &count])
```

函数的第一个参数 search 表示要被替换的目标字符串，第二个参数 replace 表示替换后的新字符串，第三个参数 subject 表示原始字符串。其中第 4 个参数是 PHP 5.0 新加入的，它包含了被替换的次数。



提示　在 PHP 4.0.5 中，可以以数组的方式传递前面的 3 个参数。如果原始字符串 subject 是一个数组，则该函数会依次用 replace 替换 subject 数组中每个元素中的 search 子字符串，同时该函数的返回值也是一个数组。

如果 search 是一个数组，而 replace 是一个字符串，则该函数将会用 replace 替换 search 数组中的所有成员。而如果 search 和 replace 都是数组，则会用数组 replace 的元素替换数组 search 对应的成员元素，即用 replace(0)替换 search(0)，而如果 replace 的元素少于 search 的元素，则 search 剩下的元素将会用空字符串进行代替。

实例 3-25　字符串替换

本实例演示了各种字符串的替换方法和技巧，如代码 3-25 所示。

【代码 3-25】　str_replace.php

```
<?php
// 原始字符串: <body text='black'>
// 下面语句表示用 black 替换 %body%
$bodytag = str_replace("%body%", "black", "<body text='%body%'>");
$a=htmlspecialchars("<body text='%body%'>");
$b=htmlspecialchars($bodytag);
echo "原始字符串：$a <br>";
echo "替换后：$b <br>";
// 将数组中设定的元素替换成空字符
$vowels = array("a", "e", "i", "o", "u", "A", "E", "I", "O", "U");
$onlyconsonants = str_replace($vowels, "", "Hello World of PHP");
echo "原始字符串：Hello World of PHP <br>";
```

```
echo "替换后：$onlyconsonants<br>";
//在字符串中查找一个数组中设定的内容
//然后用另外一个数组中的内容替换
$phrase  = "You should eat fruits, vegetables, and fiber every day.";
$healthy = array("fruits", "vegetables", "fiber");
$yummy   = array("pizza", "beer", "ice cream");
$newphrase = str_replace($healthy, $yummy, $phrase);
echo "原始字符串：You should eat fruits, vegetables, and fiber every day. <br>";
echo "替换后：$newphrase<br>";
// 返回替换的数量
$str = str_replace("ll", "", "good golly miss molly!", $count);

echo "被替换的数量：$count";
?>
```

执行结果如图 3-12 所示。

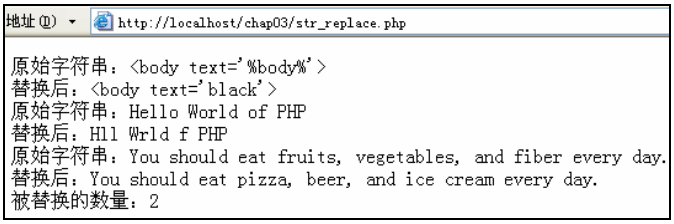


图 3-12　字符串替换

（2）substr_replace ()函数。该函数同 str_replace 函数的作用基本类似，只是该函数增加了限制条件，将用户原始字符串中的部分子字符串进行查找和替换。该函数的声明如下：

```
string substr_replace (string string, string replacement, int start [, int length])
```

第一个参数 string 表示原始字符串，第二个参数 replacement 表示替换后的新字符串，第三个参数 start 表示要被替换的目标字符串的起始位置，第四个参数 length 为可选参数，表示被替换的目标字符串的长度，如果该参数被省略，则表示从起始位置之后的所有字符串被替换。



参数 start 和 length 可以表示负数。如果 start 为正数，则表示从字符串的开始处计算，如果是一个负数，则从末尾开始的一个偏移量。length 如果为正数，则表示从 start 开始的被替换字符串的长度。如果为负数，则表示从原始字符串末尾开始到第 length 个字符串停止替换。

实例 3-26　在子字符串中替换

本实例演示了如何在一个设定的子字符串中进行查询，如代码 3-26 所示。

【代码 3-26】　substr_replace.php

```
<?php
$var = 'ABCDEFGH:/MNRPQR/';
echo "原始字符串 : $var<hr />\n";
/* 下面两句替换整个字符串 */
echo substr_replace($var, 'bob', 0) . "<br />";
echo substr_replace($var, 'bob', 0, strlen($var)) . "<br />";
```

```
/* 在句首插入字符串，即被替换的字符串为空 */
echo substr_replace($var, 'bob', 0, 0) . "<br />";
/* 下面两句用'bob'替换'MNRPQR'  */
echo substr_replace($var, 'bob', 10, -1) . "<br />";
echo substr_replace($var, 'bob', -7, -1) . "<br />";
/* 删除'MNRPQR' */
echo substr_replace($var, '', 10, -1) . "<br />";
?>
```

执行结果如图 3-13 所示。

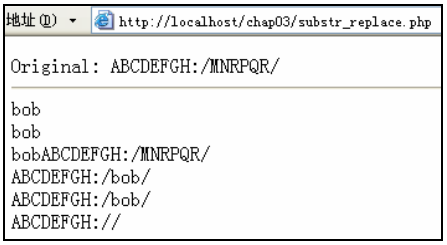


图 3-13　在子字符串中替换

（3）str_ireplace()函数。该函数同 str_replace 函数一样，只是该函数对大小写不敏感。函数声明如下：

```
mixed str_ireplace ( mixed search, mixed replace, mixed subject [, int &count])
```

应用如下面代码所示：

```
<?php
$bodytag = str_ireplace("%body%", "black", "<body text=%BODY%>");
echo htmlspecialchars($bodytag);
?>
```

输出结果如下：

```
<body text=black>
```

3.5　正则表达式

正则表达式是一种依赖其自身的微型语言——专为用文本描述模式而设计的一种语言。尽管正则表达式是极其简洁的，但是它有自己的语法和规则——非常灵活和强大。从抽象层面上看，一个正则表达式由正则字符（字母 a、逗号、数值等）和元字符（如星号、圆括号等）所构成的，其中元字符联合构成了一个文本模式的程序性描述。因此正则表达式允许编程人员匹配文本内的模式和子模式。例如编程人员可以在单词内查找每个元音字母，从电子邮件中抽取用户名或者在网页内容中查找所有的超级链接。

PHP 中有两种形式的正则表达式，而且形式十分相似。这两种形式分别被称作“POSIX 扩展”正则表达式和“Perl 兼容”正则表达式。它们的语法稍微不同，但每种形式的基本构建是一样的。本节将介绍如何使用 Perl 兼容正则表达式，因为这种形式更加强大，而且是二进制安全的。读者可以在下列网址中获取到 POSIX 扩展正则表达式的描述：http://www.php.net/manual/en/ref.regex.php。

一旦掌握了 Perl 兼容正则表达式，则使用 POSIX 扩展正则表达式会十分容易。

正则表达式在数据有效性检查方面有着重要的作用。使用正则表达式执行简单的数据有效性检查，可以保证从网络上所接受到的任何数据都得到了正确的格式化和有效性检查。

3.5.1 正则表达式基础

正则表达式用来查找文本中的模式。正则表达式用特殊的定界字符，通常用正斜杠（/）将其两端括起来。正则表达式内有两种明确类型的字符：正则字符和元字符。如果使用一个没有元字符的正则表达式，则只进行纯文本搜索；如果完全不采取正则表达式，使用 strstr() 函数也可实现同样的效果。使用元字符，可以把字符（和其他元字符）组合成匹配于各种复杂模式的更高级表达式。每个正则表达式都是由表达式组成，而表达式可以像单个字符一样简单，也可以像含有多层子表达式的复合结构那样复杂。

例如，正则表达式“/dog/”将匹配串“dog”，这也是一个没有使用元字符的正则表达式。相反，正则表达式“/[dog]/”使用了元字符“[”和“]”，因此将执行匹配串“d”、串“o”或者“g”的更复杂的工作。

正则表达式中的任意一个正则字符都匹配它自身。正则表达式中的 P 将匹配串中的 P。这样的匹配在默认的情况下是区分大小写的。

下面介绍正则表达式中的元字符，如表 3-3 所示。

表 3-3 正则表达式的元字符

元 字 符	名 称	描 述
\	反斜杠	转义字符。如\t 将匹配一个制表符，而不是匹配字母 t
\|	管道运算符	表示“匹配其中任何一个”，如 dog\|pig 将匹配 dog 或者 pig
()	开始与结束圆括号	把字符、元字符和子表达式组合成组，如(foo\|bar)baz 将匹配 fool 或者 bar 后跟 baz
[]	开始与结束方括号	建立一个字符类，该类将匹配方括号内所包括的任意一个字符（或者表达式），例如[xyz]将匹配任何一个这样的串，只要它含有字母 x、y 和 z 中的任意一个
{ }	开始与结束大括号	为前导表达式定义一个最小/最大匹配数目，例如 a{2,4}将匹配串 aa,aaa 和 aaaa
^	插入符	匹配一行的起点，或者说，如果^是一个字符类中的第一个字符，意味着“匹配这个字符类中没有被发现的任何字符”
$	美元符	匹配一行的结束点
*	星号	匹配 0 个或者多个前导表达式
?	问号	匹配 0 个或者 1 个前导表达式
+	加号	匹配 1 个或者多个前导字符表达式
.	句点或者点	匹配任一字符

正则表达式中的元字符不被翻译成正常字符。如果希望这些字符之一在正则表达式中用做实际值，必须用一个反斜杠来转义（即前缀）。假如要查找一个点，则需要在模式内搜索“\.”。

正则表达式在正常的情况下包含在两个斜杠（即正斜杠）之间，其中一个在正则表达式的开始处，另一个在结束处（尽管可以使用任何一个字符作为界定符，但是“/”是一个公共约定）。因此一个最基本的正则表达式如“/php/”。

3.5.2 匹配正则表达式模式

preg_match 函数用来匹配一个正则表达式。该函数的声明如下所示：

```
int preg_match (string pattern, string subject [, array matches [, int flags]])
```

该函数在 subject 字符串中搜索与 pattern 给出的正则表达式相匹配的内容。如果提供了第三个参数 matches，则其会被搜索的结果所填充。$matches[0]将包含与整个模式匹配的文本，$matches[1]将包含与第一个捕获的括号中的子模式所匹配的文本，依此类推。

flags 可以是下列标记。

PREG_OFFSET_CAPTURE：如果设定本标记，对每个出现的匹配结果也同时返回其附属的字符串偏移量。注意这改变了返回的数组的值，使其中的每个单元也是一个数组，其中第一项为匹配字符串，第二项为其偏移量。

preg_match()返回 pattern 所匹配的次数。要么是 0 次（没有匹配）或 1 次，因为 preg_match()在第一次匹配之后将停止搜索。preg_match_all()则相反，会一直搜索到 subject 的结尾处。如果出错，preg_match()返回 FALSE。

提示　如果只想查看一个字符串是否包含在另一个字符串中，不需要用 preg_match()，用 strpos()或 strstr()替代，要快得多。

实例 3-27　在文本中搜索“PHP”字符串

本实例演示如何通过 preg_match 搜索目标字符串，如代码 3-27 所示。

【代码 3-27】　preg_match_1.php

```
<?php
// 模式定界符后面的 "i" 表示不区分大小写字母的搜索
if (preg_match ("/php/i", "PHP is very good.")) {
    print "找到匹配.";
} else {
    print "没找到匹配.";
}
?>
```

执行结果如下：

```
找到匹配.
```

实例 3-28　在文本中搜索“Web”字符串

本实例演示如何通过正则表达式搜索单词“web”，如代码 3-28 所示。

【代码 3-28】　preg_match_2.php

```
<?php
/* 模式中的 \b 表示单词的边界，因此只有独立的 "web" 单词会被匹配，
* 而不会匹配例如 "webbing" 或 "cobweb" 中的一部分 */
if (preg_match ("/\bweb\b/i", "PHP is  web scripting language .")) {
    print "找到匹配<br>";
} else {
    print "没找到匹配<br>";
}

if (preg_match ("/\bweb\b/i", "PHP is the website scripting language.")) {
    print "找到匹配<br>";
} else {
    print "没找到匹配<br>";
}
?>
```

执行结果如下：

```
找到匹配
没找到匹配
```

实例 3-29　从 URL 中取出域名

本实例演示了如何从 URL 中取出域名，如代码 3-29 所示。

【代码 3-29】　preg_match_3.php

```
<?php
// 从 URL 中取得主机名
preg_match("/^(http:\/\/)?([^\/]+)/i",
    "http://www.php.net/index.html", $matches);
//获取主机名
$host = $matches[2];

// 从主机名中取得后面两段得到域名
preg_match("/[^\.\/]+\.[^\.\/]+$/", $host, $matches);
echo "域名为: {$matches[0]}\n";
?>
```

执行结果如下：

```
域名为: php.net
```

3.5.3　锚定一个匹配

（1）插入符。插入符（^）是两个锚定元字符之一。这个元字符使正则表达式匹配本行起始处所出现的字符。为了使得正则表达式“/^PHP/”在某个串中成功地找到一个匹配，字母 PHP 必须在一行的起始处。这些字符后面跟着什么内容没关系，只要前面 3 个字符是“PHP”就行。

实例 3-30　插入符“ˆ”的应用

本实例演示插入符的应用，如代码 3-30 所示。

【代码 3-30】　preg_match_4.php

```
<?php
$str="PHP is the best scripting language";
$pattern="/^PHP/";
if (preg_match($pattern,$str)
    print("发现匹配");
?>
```

执行结果如下：

```
发现匹配
```

在上面代码中，由于字符串 str 的头 3 个字符是 PHP，所以模式是匹配的。如果将 str 改成“I think php is the best scripting language”，结果则是不匹配的。

（2）美元符。美元符（$）是把一个模式锚定到一行的尾端（或串的尾端）。

实例 3-31　美元符“$”的应用

本实例演示美元符的应用，如代码 3-31 所示。

【代码 3-31】　preg_match_5.php

```
<?php
$str="I like PHP";
$pattern="/PHP$/";
if (preg_match($pattern,$str))
    print("发现匹配!");
?>
```

执行结果如下：

```
发现匹配!
```

以上输出结果证明是匹配的，因为字符串最后的 3 个字符是 PHP。

另外，可以使用这两个元字符来锚定任何一个模式，可以彼此联合用来保证将该模式锚定到一行的起点和终点。

实例 3-32　美元符“$”和插入符“^”的联合使用

本实例演示美元符和插入符的联合使用，如代码 3-32 所示。

【代码 3-32】　preg_match_6.php

```
<?php
$pattern="/^PHP$/";
$str="PHP";
if (preg_match($pattern,$str))
    print("发现匹配!<br>");
else
    print("没发现匹配!<br>");

$str=" PHP is the best scripting language";
if (preg_match($pattern,$str))
    print("发现匹配!<br>");
else
    print("没发现匹配!<br>");
?>
```

执行结果如下：

```
发现匹配!
没发现匹配!
```

结果是第一个匹配成功。第二个匹配失败，因为第二个字符串并不是以 PHP 结束的。

3.5.4　匹配单个字符

最基本的正则表达式是匹配其自身的单个字符，比如表达式 PHP 中的字母 P。本节将介绍一个十分有用的元字符“.”（点），意思是“匹配除换行符之外的任一字符”。

实例 3-33　元字符“.”的应用

本实例演示如何通过元字符“.”来匹配单个字符，如代码 3-33 所示。

【代码 3-33】　preg_match_7.php

```
<?php
```

```
$pattern="/P.P/";
$str="I like PHP mostly";
if (preg_match($pattern,$str))
    print("发现匹配!");
?>
```

执行结果如下：

```
发现匹配!
```

从输出结果可以看出以上模式是匹配的。因为该匹配需要在字符串中先找到 P，然后找到任一字符，然后再找到一个 P，因为字符串中含有 PHP，所以是匹配的。

3.5.5 匹配字符类

一个字符类就像正则表达式内部的一种微型语言，表示与一组方括号（[]）内所定义的那个组的某部分相匹配的任何一个字符。当需要查找两个字符中的任何一个都适合的串时，使用字符类方括号最为方便。如模式“/c[oa]t”可以匹配 cat，也可以匹配 cot。

字符类中的元字符集与正规表达式中的普通列表稍有不同。字符类中最重要的元字符之一是短横线（-），该元字符被称为范围界定符。可以在一个字符类中指定一个字符范围，如指定[2-6]来取代[23456]。



如果想匹配短横线自身，则必须在短横线前面加一个反斜线，如[1-9\-]将匹配任何数字或者短横线。

一个字符类可以包含多个范围。如果只需要匹配字母和数字字符，可以使用模式“a-zA-Z0-9”。

如果需要匹配不在某个字符集内的任意字符，可以使用否定字符类。在一个字符类中，插入符（^）表示一个否定字符类（并不是前面提到的“行开始”普通的正则表达式含义）。

实例 3-34 匹配字符类

本实例演示了使用短横线和插入符来匹配字符类，如代码 3-34 所示。

【代码 3-34】 preg_match_8.php

```
<?php
$pattern="/[^A-Z]$/";
$str="I like PHP";
if (preg_match($pattern,$str))
    print("发现一个匹配!");
else
    print("没发现一个匹配!");
?>
```

执行结果如下：

```
没发现一个匹配!
```

以上结果会显示不匹配，因为在字符串“I like PHP”中，是以 P 结束的，而模式要求不能以大写的“A”到“Z”作为结束字符。



字符类的插入符“^”将不作为否定元字符，而是仅仅作为一个普通字符“^”进行匹配。如“/[a^b]”将匹配 a、b 或者^。

3.5.6　替换匹配

本节将要介绍另外一个元字符，即管道元字符（|）。管道元字符在正则表达式中的意思是“或者（OR）”。通过管道元字符，可以匹配管道元字符左边或者右边的每个字符串直到该表达式的结尾。

实例 3-35　替换匹配

本实例演示管道元字符“|”的应用，如代码 3-35 所示。

【代码 3-35】　preg_match_9.php

```
<?php
$pattern="/(dog|cat)\.$/";
$str="I like dog.";
if (preg_match($pattern,$str))
    print("发现匹配!");
?>
```

执行结果如下：

```
发现匹配!
```

从以上代码可以看出是匹配的。因为正则表达式“/(dog|cat)\.$”要匹配的是以“cat.”或者以“dog.”结束的字符串，而字符串“I like dog.”以“dog.”结束，所以匹配。

提示　在上面代码中使用了圆括号，表示是子表达式。一个子表达式是一个包含在圆括号内的表达式的任意部分，其中圆括号用来定义一个操作应该适用的范围。

3.5.7　不区分大小写的模式

读者可能已经注意到，正则表达式是区分大小写的。尽管使用字符类可以轻松解决这个问题，如[Aa][Bb][Cc]，但是还有一种更为简单的方法告诉 preg_match 函数忽略大小写。这种方法就是在正则表达式的最后正斜杠后面添加一个字符“i”，比如/cat/i 这个表达式将匹配任意一个可能的大小写组合，如 Cat、cat、CAt 等。

3.5.8　分组模式

当需要匹配“123-4567”这样的字符串的时候，分组模式就比较重要。分组模式在正则表达式中用 3 个字符来建立：星号（*）、问号（?）和加号（+）。它们的意思分别是匹配“0 个或者多个”、“0 个或者 1 个”、“1 个或者多个”前导字符。

下面列举一些分组模式的例子。

（1）/do*g/：该表达式将匹配字母 d 后面跟任意个数的字符 o，最后再跟一个字符 g，如 dg、dog、doog、doooood 等。

（2）/do?g/：该表达式将匹配字母 d 后面跟一个字母 o 然后跟 g，或者字母 d 后面直接跟 g，即 dog 或者 dg，不匹配 doog、dooog 等。

（3）/do+g/：该模式将匹配字母 d 后面跟一个或者多个字母 o，最后跟字母 g，如 dog、doog、dooog 等，但是不匹配 dg。

提示 分组模式除了可以应用到单个字符中，还可以应用到任何一个子表达式中。

3.5.9 预定义字符和字符类

很多常用的字符类以及数个字符类都事先被定义成简写的形式，可以方便编程人员使用。比如可以使用“\d”来代替字符类“[0-9]”来指定任一数字。

预定义字符和字符类如表 3-4 所示。

表 3-4 预定义字符和字符类

字 符	含 义	字 符	含 义
\a	蜂鸣	\D	非数字字符（和[^0-9]相同）
\n	换行符	\w	单词字符（和[a-zA-Z0-9_]相同）
\r	回车	\W	非单词字符
\t	制表符	\s	空格字符（和[\t\n\r\f]相同）
\f	走纸	\S	非空格字符
\e	专义	\xX	十六进制数字
\d	数字（和[0-9]相同）	\cX	匹配相应的控制字符

3.5.10 限定符

除了前面介绍的分组模式外，还可以指定想要的精确重复次数。可以使用大括号（{}）指出限定符。表 3-5 所示列出了可供使用的所有限定符。

表 3-5 限定符语法

限 定 符	最 小 值	最 大 值	范 围
{m}	m	m	精确的 m 次
{m,x}	m	x	至少 m 次，最多 x 次
{m,}	m	无限大	至少 m 次
*	0	无限大	0 次或者多次
?	0	1	0 次或者 1 次
+	1	无限大	1 次或者多次

{}限定符用于匹配更精确个数的字符。借助一个使用了{}限定符的简单正则表达式，可以很轻松地匹配一个带有或者不带有区号的电话号码：

```
/^(\d{3}[\.\-]\d{4}|\d{3}[\.\-]\d{3}[\.\-]\d{4})$/
```

下面对以上正则表达式简要说明。

^：表示串或者行的开始。

\d{3}：3 个连续的数字。

[\.\-]：一个单独的点或者一个单独的短横线。

\d{4}：4 个连续的数字。

|：管道元字符，表示忽略前面的一切，然后重新匹配。

\d{3}：3 个连续的数字。

[\.\-]：一个单独的点或者一个单独的短横线。

\d{4}：另外的 4 个连续数字。

$：串或者行的结束。

下面的一些号码都是与以上正则表达式匹配的：123.456.7890，123-456-7890。

3.5.11　逆向引用

一个逆向引用是用反斜杠转义的数，该数指出当前表达式应该再次匹配它已经查找的某个序列。此时，圆括号相当于存储器，逆向引用的数目指出当前表达式括起的部分应该重用，其中第一个括起部分是\l。

3.5.12　其他常用正则函数

下面介绍 PHP 中一些常用的正则表达式函数，通过这些函数，可以轻松地通过正则表达式来完成一些功能，如字符串的查找和替换等。

（1）preg_replace()函数。该函数执行正则表达式的搜索和替换。函数声明如下：

```
mixed preg_replace ( mixed pattern, mixed replacement, mixed subject [, int limit])
```

该函数在 subject 中搜索 pattern 模式的匹配项并替换为 replacement。如果指定了 limit，则仅替换 limit 个匹配，如果省略 limit 或者其值为–1，则所有的匹配项都会被替换。

replacement 可以包含\\n 形式或$n 形式的逆向引用，首选使用后者。每个此种引用将被替换为与第 n 个被捕获的括号内的子模式所匹配的文本。n 可以从 0～99，其中\\0 或$0 指的是被整个模式所匹配的文本。对左圆括号从左到右计数（从 1 开始）以取得子模式的数目。

对替换模式在一个逆向引用后面紧接着一个数字时（即紧接在一个匹配的模式后面的数字），不能使用熟悉的\\1 符号来表示逆向引用。举例说“\\11”，将会使 preg_replace()搞不清楚是想要一个“\\1”的逆向引用后面跟着一个数字“1”还是一个“\\11”的逆向引用。本例中的解决方法是使用“\${1}1”。这会形成一个隔离的$1 逆向引用，而使另一个“1”只是单纯的文字。

可能理解以上说明比较吃力，下面通过实例来学习。

下面的代码说明了逆向引用后面紧接着数字的用法。

```
<?php
$string = "April 15, 2003";
$pattern = "/(\w+) (\d+), (\d+)/i";
$replacement = "\${1}1,\$3";
print preg_replace($pattern, $replacement, $string);

?>
```

输出结果如下：

```
April1,2003
```

如果搜索到匹配项，则会返回被替换后的 subject，否则返回原来不变的 subject。

preg_replace()的每个参数（除了 limit）都可以是一个数组。如果 pattern 和 replacement 都是数组，将以其键名在数组中出现的顺序来进行处理，不一定和索引的数字顺序相同。如果使用索引来标识哪个 pattern 将被哪个 replacement 来替换，应该在调用 preg_replace()之前用 ksort()对数组进行排序。

实例 3-36 数组替换

本实例演示利用 preg_replace 函数进行数组替换，如代码 3-36 所示。

【代码 3-36】 preg_replace_1.php

```
<?php
$string = "The quick brown fox jumped over the lazy dog.";
$patterns[0] = "/quick/";
$patterns[1] = "/brown/";
$patterns[2] = "/fox/";

$replacements[2] = "bear";
$replacements[1] = "black";
$replacements[0] = "slow";
print preg_replace($patterns, $replacements, $string);
echo "<br>";
//对字符串进行键值排序
ksort($patterns);
ksort($replacements);
print preg_replace($patterns, $replacements, $string);
?>
```

执行结果如下：

```
The bear black slow jumped over the lazy dog.
The slow black bear jumped over the lazy dog.
```

如果 subject 是数组，则会对 subject 中的每个项目执行搜索和替换，并返回一个数组。

如果 pattern 和 replacement 都是数组，则 preg_replace()会依次从中分别取出值来对 subject 进行搜索和替换。如果 replacement 中的值比 pattern 中的少，则用空字符串作为余下的替换值。如果 pattern 是数组而 replacement 是字符串，则对 pattern 中的每个值都用此字符串作为替换值。反过来则没有意义了。

/e 修正符使 preg_replace()将 replacement 参数当作 PHP 代码（在适当的逆向引用替换完之后）。

提示 要确保 replacement 构成一个合法的 PHP 代码字符串，否则会在包含 preg_replace()的行中出现语法解析错误。

实例 3-37 将 HTML 替换成文本

本实例演示了如何通过 preg_replace 函数将 HTML 替换成文本，如代码 3-37 所示。

【代码 3-37】 preg_replace_2.php

```
<?php
// $document 应包含一个 HTML 文档。
// 本例将去掉 HTML 标记，javascript 代码
// 和空白字符。还会将一些通用的
// HTML 实体转换成相应的文本。

$search = array ("'<script[^>]*?>.*?</script>'si",  // 去掉 javascript
```

```
                "'<[\/\!]*?[^<>]*?>'si",          // 去掉 HTML 标记
                "'([\r\n])[\s]+'",                // 去掉空白字符
                "'&(quot|#34);'i",                // 替换 HTML 实体
                "'&(amp|#38);'i",
                "'&(lt|#60);'i",
                "'&(gt|#62);'i",
                "'&(nbsp|#160);'i",
                "'&(iexcl|#161);'i",
                "'&(cent|#162);'i",
                "'&(pound|#163);'i",
                "'&(copy|#169);'i",
                "'&#(\d+);'e");                   // 作为 PHP 代码运行

$replace = array ("",
                  "",
                  "\\1",
                  "\"",
                  "&",
                  "<",
                  ">",
                  " ",
                  chr(161),
                  chr(162),
                  chr(163),
                  chr(169),
                  "chr(\\1)");

$text = preg_replace ($search, $replace, $document);
?>
```

（2）preg_split()函数。该函数用正则表达式分割字符串。函数声明如下：

```
array preg_split ( string pattern, string subject [, int limit [, int flags]])
```

该函数返回一个数组，包含 subjec 中沿着与 pattern 匹配的边界所分割的子串。如果指定了 limit，则最多返回 limit 个子串，如果 limit 是–1，则意味着没有限制，可以用来继续指定可选参数 flags。

flags 可以是下列标记的任意组合（用按位或运算符 | 组合）。

PREG_SPLIT_NO_EMPTY：如果设定了本标记，则 preg_split()只返回非空的成分。

PREG_SPLIT_DELIM_CAPTURE：如果设定了本标记，定界符模式中的括号表达式也会被捕获并返回。

PREG_SPLIT_OFFSET_CAPTURE：如果设定了本标记，对每个出现的匹配结果也同时返回其附属的字符串偏移量。注意，这改变了返回的数组的值，使其中的每个单元也是一个数组，其中第一项为匹配字符串，第二项为其在 subject 中的偏移量。

实例 3-38　将字符串分割成字符

本实例演示了如何通过 preg_split 函数将字符串分割成字符，如代码 3-38 所示。

【代码 3-38】　preg_split_1.php

```
<?php
```

```
// 分隔字符串
// 分隔符包括 " ", \r, \t, \n and \f
$keywords = preg_split ("/[\s,]+/", "hypertext language, programming");
print_r($keywords);
?>
```

执行结果如下：

```
Array ( [0] => hypertext [1] => language [2] => programming )
```

从结果可以看出，字符串被分隔成 3 个单词，保存在数组中。

实例 3-39　将字符串分割成匹配项和偏移量

本实例演示了如何将字符串分割为匹配项及其偏移量，如代码 3-39 所示。

【代码 3-39】　preg_split_2.php

```
<?php
$str = 'hypertext language programming';
$chars = preg_split('/ /', $str, -1, PREG_SPLIT_OFFSET_CAPTURE);
print_r($chars);
?>
```

输出结果如下：

```
Array
(
    [0] => Array
        (
            [0] => hypertext
            [1] => 0
        )

    [1] => Array
        (
            [0] => language
            [1] => 10
        )

    [2] => Array
        (
            [0] => programming
            [1] => 19
        )

)
```

有关正则表达式的应用，就介绍到这里。本节内容有一些难度，希望读者能够结合实例多练习。

第 4 章　PHP 与 HTML 表单

从本章开始，读者将会陆续学习到利用 PHP 开发动态网站的技术。

既然是网络技术，其最基本的 HTML 知识也是不可缺少的，本书不专门介绍 HTML，只是介绍和 PHP 编程比较相关的一些基础知识。

4.1　HTML 表单

表单是 HTML 中的一个重要内容。大部分的网页交互都是通过表单来实现的。在一个标准的 HTML 文件中，定义一个表单非常简单，只需要用下面的关键字来实现：

```
<form name="" method="" action="" enctype="">
    //表单元素
</form>
```

其中<form></form>就是表单定义的关键字。后面的几个参数代表不同的含义：

（1）action：该属性包含脚本的 URL，这个脚本处理通过表单提交的信息。提交信息也就是把用户输入的信息发送到 Web 服务器以进行处理。

（2）enctype：该属性用于为特殊的协议指定某些格式，从而可以将信息传送到服务器上。

（3）methord：该属性为表单的提交方式。通常有 GET 和 POST 两种方式。

（4）name：该属性为表单的名称，用于区分不同的表单。

如果要定义一个基本的用户资料调查的表单，可以使用代码 4-1 所示的 HTML 文件。

【代码 4-1】　firsthtml.htm

```
<html>
<head>
<title>
用户调查表
</title>
</head>
<body>
欢迎光临本网站，请首先输入以下个人资料：<br>
<form method=post action="showdetail.php">
账号：<INPUT maxLength=25 size=16 name=login ><br>
姓名：<INPUT type=password size=19 name=yourname ><br>
密码：<INPUT type=password size=19 name=passwd ><br>
确认密码：<INPUT type=password size=19 name=passwd ><br>
查询密码问题：<br>
<select name=question>
    <option selected value="">--请您选择--</option>
```

```
        <option value="我的宠物名字？">我的宠物名字？</option>
        <option value="我最好的朋友是谁？">我最好的朋友是谁？</option>
        <option value="我最喜爱的颜色？">我最喜爱的颜色？</option>
        <option value="我最喜爱的电影？">我最喜爱的电影？</option>
        <option value="我最喜爱的影星？">我最喜爱的影星？</option>
        <option value="我最喜爱的歌曲？">我最喜爱的歌曲？</option>
        <option value="我最喜爱的食物？">我最喜爱的食物？</option>
        <option value="我最大的爱好？">我最大的爱好？</option>
    </select>

    <br>
    查询密码答案：<input name=question2 size=18><br>

    出生日期：
            <select name="byear" id="BirthYear" tabindex=8>
                <script language="JavaScript">
                    var tmp_now = new Date();
                    for(i=1930;i<=tmp_now.getFullYear();i++){
            document.write("<option value='"+i+"' "+(i==tmp_now.getFullYear()-
    24?"selected":"")+">"+i+"</option>")
                    }
                    </script>
            </select>
             年
             <select name="bmonth">
            <option value="01" selected>1</option>
            <option value="02">2</option>
            <option value="03">3</option>
            <option value="04">4</option>
            <option value="05">5</option>
            <option value="06">6</option>
            <option value="07">7</option>
            <option value="08">8</option>
            <option value="09">9</option>
            <option value="10">10</option>
            <option value="11">11</option>
            <option value="12">12</option>
             </select>
             月
        <select name=bday tabindex=10  alt="日:无内容">
            <option value="01" selected>1</option>
            <option value="02">2</option>
            <option value="03">3</option>
            <option value="04">4</option>
            <option value="05">5</option>
            <option value="06">6</option>
            <option value="07">7</option>
            <option value="08">8</option>
            <option value="09">9</option>
            <option value="10">10</option>
            <option value="11">11</option>
            <option value="12">12</option>
```

```
            <option value="13">13</option>
            <option value="14">14</option>
            <option value="15">15</option>
            <option value="16">16</option>
            <option value="17">17</option>
            <option value="18">18</option>
            <option value="19">19</option>
            <option value="20">20</option>
            <option value="21">21</option>
            <option value="22">22</option>
            <option value="23">23</option>
            <option value="24">24</option>
            <option value="25">25</option>
            <option value="26">26</option>
            <option value="27">27</option>
            <option value="28">28</option>
            <option value="29">29</option>
            <option value="30">30</option>
            <option value="31">31</option>
        </select>
    <br>
    性别：<input type="radio" name="gender" value="1" checked>
         男
         <input type="radio" name="gender" value="2" >
         女
    <br>
    请选择你的爱好：<br>
    <input type="checkbox" name="hobby[]" value="dance" >跳舞<br>
    <input type="checkbox" name="hobby[]" value="tour" >旅游<br>
    <input type="checkbox" name="hobby[]" value="sing" >唱歌<br>
    <input type="checkbox" name="hobby[]" value="ball" >打球<br>
    <input type="submit" name="submit" value="提交">
    <input type="reset" name="reset" value="重填">
    <br>
    </body>
    <html>
```

执行结果如图 4-1 所示。

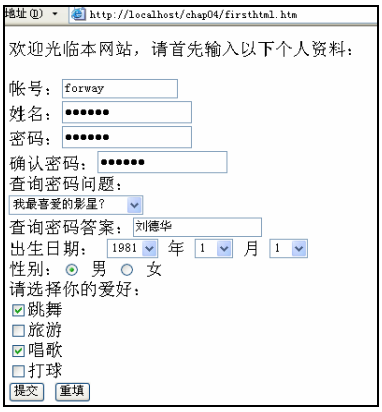


图 4-1　代码 4-1 的执行结果

图 4-1 所示显示了 HTML 中的常用表单，当用户输入数据或者做出选择之后，单击“提交”按钮，将数据提交到服务器，然后服务器显示出用户的资料或者将资料加入到数据库中。但是客户端浏览器是如何将数据传递到服务器，下面就需要解释浏览器提交数据的方法。

在代码 4-1 中，表单<form>的定义格式如下：

```
<form method=post action="showdetail.php">
```

其中“method”表示的是数据提交的方法。最常用的方法有 POST 方法和 GET 方法；“action”表示当用户单击了“提交”按钮后，需要进行数据处理的文件，也就是说在代码 4-1 中，当用户单击“提交”按钮后，会连接到 showdetail.php 文件，并将数据传递给该文件，让该文件进行各种后续处理。

4.1.1 GET 提交方法

GET 提交方法本质上是将数据通过连接地址的形式传递到下一个页面。实现 GET 提交方法有两种途径，一种就是通过表单的形式，另外一种是通过直接书写超级链接的形式。

提示 通过 GET 提交的数据一般数据量比较小，如果是大量数据，则尽量采用 POST 方法进行提交。

（1）通过超级链接。如果没有数据传递给服务器，则可以直接输入 URL 地址，则浏览器默认将其作为 GET 方法提交，如下所示：

```
http://localhost/chap04/firsthtml.htm
```

如果此时有数据传递给服务器端，则可以采用以下方法：

```
http://localhost/chap04/firsthtml.htm?name1=value1&name2=value2...
```

从上面可以看到，通过 GET 方法传递数据的时候，只需要将传递的数值放在 URL 地址后面，通过问号“?”分隔开。传递数据的格式就是“name=value”，name 就是要传递数据的名称，而 value 是要传递的数值，其中 name 主要是能够让服务器识别。当有多个值要传递的时候，多个值之间可以通过符号“&”分隔开。

这种方法多用在超级链接中，当传递数据不多的时候，可以直接通过链接来传递数据。接下来介绍一种通过表单传递数据的方法。

（2）通过表单传递数据。通过 GET 方法提交数据，书写格式同代码 4-1 比较类似，只是由于 GET 方法提交的数据量不宜太大，所以下面简化了其代码。

实例 4-1 通过 GET 方法提交数据

本实例演示了如何通过 GET 方法提交数据。实例由两个文件组成，一个为 HTML 表单文件，文件名为 get.htm，如代码 4-2 所示，另一个文件为表单处理文件，文件名为 get.php，如代码 4-3 所示。

【代码 4-2】 get.htm

```
<html>
<head>
<title>
用户调查表
</title>
</head>
```

```
<body>
请输入账号和密码：<br>
<form method=get action="get.php">
账号：<INPUT maxLength=25 size=16 name=login ><br>
密码：<INPUT type=password size=19 name=passwd ><br>
<input type="submit" name="submit" value="提交">
<input type="reset" name="reset" value="重填">
<br>
</body>
<html>
```

表单的显示界面如图 4-2 所示。



图 4-2　简单表单

在代码 4-2 中，<form>表单的定义如下：

```
<form method=get action="get.php">
```

其中“method”即提交方法被定义成“GET”，表示要用 GET 方法提交。“action”被定义成“get.php”，表示提交后的处理文件为“get.php”。

【代码 4-3】　get.php

```
<html>
<head>
<title>
get 提交
</title>
</head>
<body>
我是通过 GET 方法提交而来的哦！
</body>
<html>
```

提示　由于还没有介绍如何通过 PHP 来获取提交的数据，因此现在只是简单的输出一个提交完成信息。

当用户输入数据以后，单击“提交”按钮后，出现的界面如图 4-3 所示。



图 4-3　GET 方法提交后的结果

这里重点观察地址栏中的 URL，当用户单击“提交”按钮后，地址栏变成了“http://localhost/chap04/get.htm?login=forway&passwd=forway&submit=%CC%E1%BD%BB”，这表示是客户端通过该地址向服务器发送了请求，从地址栏的结构可以看出，它的本质还是将数据附加在 URL 后面传递给数据库的，而且格式也是“name=value”的形式。其中“login”、

“passwd”以及“submit”为表单的名称。

提示 读者可能奇怪后面的“submit=%CC%E1%BD%BB”内容，这是因为“submit”是“提交”按钮的名称，而内容为“提交”，但是 URL 是有一定的格式需求，因此在提交之前，“提交”两个字被进行了处理，所以出现了“%CC%E1%BD%BB”。

通过表单提交数据和直接在浏览器中输入“http://localhost/chap04/get.htm?login=forway&passwd=forway”的链接地址，得到的结果是一样的。

下面介绍另外一种提交方式，即 POST 的提交方式。

4.1.2 POST 提交方法

POST 方法提交数据，不能通过 URL 的形式来实现，只能通过表单形式来实现。在提交大量数据的时候，使用 POST 方法。这里为了便于比较，使用同 get.htm 文件基本类似的表单。

实例 4-2 通过 POST 方法提交数据

本实例演示通过 POST 方法提交数据。该实例由两个文件组成，一个是前台 HTML 文件，文件名为 post.htm，如代码 4-4 所示；另一个为提交处理文件，文件名为 post.php，如代码 4-5 所示。

【代码 4-4】 post.htm

```
<html>
<head>
<title>
用户调查表
</title>
</head>
<body>
请输入账号和密码：<br>
<form method=post action="post.php">
账号：<INPUT maxLength=25 size=16 name=login ><br>
密码：<INPUT type=password size=19 name=passwd ><br>
<input type="submit" name="submit" value="提交">
<input type="reset" name="reset" value="重填">
<br>
</body>
<html>
```

post.htm 文件的代码同 get.htm 代码基本类似，只是将提交方法改成了“POST”，其显示界面同 get.htm 的界面一样，见图 4-2。

“action”被定义成“post.php”文件，表示提交后的处理文件为“post.php”文件。

【代码 4-5】 post.php

```
<html>
<head>
<title>
get 提交
</title>
```

```
</head>
<body>
我是通过 POST 方法提交而来的哦!
</body>
<html>
```

当用户单击“提交”按钮后，结果如图 4-4 所示。



图 4-4　POST 方法提交后的界面

比较图 4-3 和图 4-4，可以发现在地址栏中，通过 POST 方法提交后，并没有将数据附加在 URL 中。

提示　通过 POST 方法提交数据，其本质是将所有的数据作为一个单独的数据块提交到服务器，并且每个字段之间会有特定的分隔符。

关于 HTML 表单的基本内容，就先介绍到这里，接下来介绍如何获取客户端提交的各种数据。

4.2　获取提交的数据

熟悉 ASP 的读者肯定了解，在 ASP 中读取客户端的数据是通过 Request 对象来实现的。而 PHP 进行数据获取不需要任何对象，处理起来非常简单。在 PHP 中，有两种方法来获取表单提交的数据。

（1）接收通过 HTML 表单元素提交的信息，自动为表单元素分配变量。这些变量的名称与 HTML 表单元素的 Name 属性是一样的。加入在 HTML 表单页面一个名称为“txtAge”的文本框，则 PHP 处理页面时会自动将变量“$txtAge”赋给该文本框。这样，只需要通过变量$txtAge 就可以访问 HTML 表单提交的数据了。无论是通过 GET 方法还是 POST 方法提交，都可以利用这种方法来获得数据。

提示　在 PHP 4.2 之后，这种使用全局变量的方法已经被自动禁止，如果想直接通过表单控件名称来获取表单数据，必须要在 php.ini 中进行设置，将 register_global 属性的值设置为 ON，而默认的设置为 OFF。

（2）将提交的数据保存在全局数组中，然后调用系统特定的自动全局变量数组来获取这些值。常用自动全局变量如下所示：

```
$_GET
$_POST
$_REQUEST = $_GET + $_POST + $_COOKIE
```

在获取表单数据中，最常用的自动全局变量是$_GET 和$_POST，它们分别获取通过 GET 方法提交的数据和通过 POST 方法提交的数据。

比如一个名称为“txtAge”的文本框表单控件，如果要获取它提交的值，则首先需要知道提交的方法，如果是 GET 方法提交，则可以用下面语句获得：

```
$_GET["txtAge"]
```

或者：

```
$_GET['txtAge']
```

如果是采用 POST 方法提交的数据，则可以使用下面的语句获得：

```
$_POST["txtAge"]
```

或者

```
$_POST['txtAge']
```

如果不知道是哪种提交方法或者不关心数据来源，则可以通过下面的语句获得：

```
$_REQUEST["txtAge"]
```

或者：

```
$_REQUEST['txtAge']
```

$_REQUEST 自动全局变量包含了所有 GET、POST、COOKIE 和 FILE 的数据，但是这样会降低处理速度，建议使用$_GET 和$_POST 方法来获取。

下面介绍常用的表单控件的数据获取方式。

4.2.1 文本框

文本框是最常用的表单元素，用于输入用户名、地址、密码等（密码类型的文本框，需要将类型设置为 password）。

下面是一个简单的文本框表单元素：

```
<INPUT type=text  maxLength=25 size=16 name=login >
```

文本框表单控件的名称为“login”，在处理页面时，可以用下面几种方法来获取：

```
$login  //直接通过全局变量来获取数据，不推荐使用
$_GET['login']  //如果通过 GET 方法提交，推荐使用
$_POST['login']   //如果通过 POST 方法提交，推荐使用
```

4.2.2 单选按钮

单选按钮表示从多个选项中选择一个。通常情况下，同一组单选按钮的名称是一样的。假如有多个单选按钮，在实际提交数据的时候，PHP 只会分配一个变量给该组单选按钮，看下面的代码：

```
<input type="radio" name="gender" value="1" checked>
<input type="radio" name="gender" value="2" >
```

以上代码创建了一组两个单选按钮，按钮的名称为“gender”，但是“value”值有两个，分别为“1”和“2”。提交后，假如用户选择了“1”，则该变量的值就为 1，如果为“2”，则变量的值也为 2，依此类推，如果一组有多个选项也一样。

要获取单选按钮的值，可以采用下面的语法：

```
$_GET["gender "]    //通过 get 方法提交
$_POST["gender "]  //通过 post 方法提交
```



提示 由于 PHP 不鼓励使用全局变量的方法，因此本书后面不会采用这种方法。

4.2.3 复选框

复选框通常能够让用户选择多个选项，比如个人爱好等。一般来说，同一组复选框按钮的名称是不一样的，但是也可以都一样。

实例 4-3　获取复选框提交的数据

本实例由两个文件组成，第 1 个为表单文件，文件名为 checkbox.htm，如代码 4-6 所示；第 2 个为提交数据获取页面，文件名为 showcheckbox.php，如代码 4-7 所示。

【代码 4-6】　checkbox.htm

```
<html>
<head>
<title>
爱好调查表
</title>
</head>
<body>
你的爱好：<br>
<form method=get action="showcheckbox.php">
<input type="checkbox" name="dance" value="dance" >跳舞<br>
<input type="checkbox" name="tour" value="tour" >旅游<br>
<input type="checkbox" name="sing" value="sing" >唱歌<br>
<input type="checkbox" name="ball" value="ball" >打球<br>

<input type="submit" name="submit" value="提交">
<input type="reset" name="reset" value="重填">
<br>
</body>
<html>
```

Checkbox.htm 的运行界面如图 4-5 所示。



图 4-5　checkbox.htm 的运行界面

以上的一组复选框，由于名称都不一样。与其他表单属性不同，即如果该复选框没有被选中，则 PHP 不会为其分配变量。因此如果直接获取它的值，系统会报错。

【代码 4-7】　showcheckbox.php

```
<?php
echo $_GET['dance'];
echo $_GET['tour'];
echo $_GET['sing'];
echo $_GET['ball'];
?>
```

同前面介绍的方法一样，因为是通过 GET 方法提交，因此用$_GET 来获取数据。因为 4 个复选框名称不一样，因此用了 4 个不同的变量来输出。在图 4-5 中，选中“旅游”和“打球”两项。然后单击“提交”按钮，执行的结果如图 4-6 所示。



图 4-6　showcheckbox.php 的执行结果

从图中可以看出，由于其中有两个表单没有提交数据，因此出现了错误。为了解决这个问题，需要判断该变量是否为空。

实例 4-4　安全获取复选框提交的数据

本实例由两个文件组成，一个为前台的文件，如代码 4-6 所示，但要将 checkbox.htm 名称改为 checkbox1.htm，然后将代码中的<form method=get action="showcheckbox.php">修改为<form method=get action="showcheckbox1.php">，其他代码不变。另一个文件为获取数据文件，文件名为 checkbox1.php，如代码 4-8 所示。

【代码 4-8】　showcheckbox1.php

```
<?php
if (!empty($_GET['dance']))
    echo $_GET['dance'] . "<br>";
if (!empty($_GET['tour']))
    echo $_GET['tour'].  "<br>";
if (!empty($_GET['sing']))
    echo $_GET['sing'] . "<br>";
if (!empty($_GET['ball']))
    echo $_GET['ball'] . "<br>";
?>
```

在浏览器中输入“http://localhost/chap04/checkbox1.htm”，显示的界面同图 4-5 一样。然后选择“旅游”和“打球”两个复选框，接着单击“提交”按钮，则会出现如图 4-7 所示的界面。

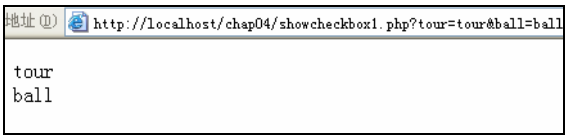


图 4-7　showcheckbox1.php 的执行结果

从图 4-7 中可以看出，代码没有错误，而且输出了用户选择的内容。但是这样处理会很复杂，需要对每个字段都要判断是否为空，接下来介绍一种快速获得复选框提交值的方法。

提示

在实际编程中，这样判断比较麻烦。因为这样的错误警告属于提示级别，不影响整个程序的运行，因此如果不想这样繁琐，可以在 php.ini 文件中，设置错误报告的级别。将 error_reporting 设置为 E_ALL & ~ (E_NOTICE)，表示会出现警告问题和错误问题，而提示问题不显示。

修改过之后，再运行实例 4-3，则不会出现提示。

实例 4-5　快速获取复选框提交的数据

快速获取复选框提交数据的方法是将所有的复选框名称统一，即所有的复选框名称相同，而值（value）设置不同。本实例由两个文件组成，一个为前台文件，用于显示表单，文件名为 checkbox2.htm，如代码 4-9 所示；另一个文件名为 showcheckbox2.php，显示提交的复选框选项，如代码 4-10 所示。

【代码 4-9】　checkbox2.htm

```
<html>
<head>
<title>
爱好调查表
</title>
</head>
<body>
你的爱好：<br>
<form method=get action="showcheckbox2.php">
<input type="checkbox" name="hobby[]" value="dance" >跳舞<br>
<input type="checkbox" name="hobby[]" value="tour" >旅游<br>
<input type="checkbox" name="hobby[]" value="sing" >唱歌<br>
<input type="checkbox" name="hobby[]" value="ball" >打球<br>


<input type="submit" name="submit" value="提交">
<input type="reset" name="reset" value="重填">
<br>
</body>
<html>
```

此处的表单名称为“hobby[]”而不是简单的“hobby”，这是因为 PHP 会自动将名称相同的表单控件提交的值保存在数组中，如果不加上“[]”，则系统默认只保存一个值。

如果要获取所有提交的数值，只需要通过 foreach 语句就可以实现了，实现代码如代码 4-10 所示。

【代码 4-10】　showcheckbox2.php

```
<?php
echo "你的爱好如下：";
if (!empty($_GET['hobby']))
foreach ($_GET['hobby'] as $hobby)
    echo($hobby . "<br>");
?>
```

showcheckbox2 的执行结果如图 4-8 所示。

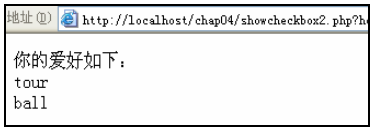


图 4-8　showcheckbox2.php 执行结果

从 showcheckbox2.php 简单的处理结果可以看出，通过将所有的复选框名称改成一样的之后，就很容易进行处理了。

4.2.4 列表框

列表框控件用于让用户进行单项或者多项选择。创建一个列表框可以通过 select 和 option 关键字来实现。通常列表框可以用来进行多选和单选。

实例 4-6 获取列表框提交的数据

本实例用于获取列表框提交的数据，由两个文件组成，前台文件名为 select.htm，如代码 4-11 所示，后台文件名为 showselect.php，如代码 4-12 所示。

【代码 4-11】 select.htm

```
<html>
<head>
<title>
选择月份
</title>
</head>
<body>
选择月份：<br>
<form method=post action="showselect.php">
        <select name="bmonth">
       <option value="01" selected>1</option>
       <option value="02">2</option>
       <option value="03">3</option>
       <option value="04">4</option>
       <option value="05">5</option>
       <option value="06">6</option>
       <option value="07">7</option>
       <option value="08">8</option>
       <option value="09">9</option>
       <option value="10">10</option>
       <option value="11">11</option>
       <option value="12">12</option>
        </select>

<input type="submit" name="submit" value="提交">
<input type="reset" name="reset" value="重填">
<br>
</body>
<html>
```

以上列表框保存了一年中的 12 个月，如果需要选择哪个月份，可以通过鼠标选择，其中第 1 个月被默认选中。但是以上的定义只能是单选的，不能多选。

代码 4-11 的执行结果如图 4-9 所示。

虽然列表框有多个选项，但是只有一个名称即“bmonth”，PHP 对单选列表框进行处理的时候，直接通过该名称来获取提交值，它的值就是被用户选中的“value”属性的值。

【代码 4-12】　showselect.php

```
<?php
    echo "你选择的月份是：<BR>";
    echo $_POST['bmonth']  . "   月";
?>
```

Showselect.php 的执行结果如图 4-10 所示。



图 4-9　select.htm 的执行结果



图 4-10　Showselect.php 的执行结果

以上介绍的是单选列表框的数据处理方法，列表框可以进行多选。在进行多项选择的时候，只需要在该元素后面添加“multiple”属性。

实例 4-7　获取列表框提交的多个选项

本实例获取列表框提交的多个选项。由两个文件组成，前台文件名为 select1.htm，如代码 4-13 所示；后台文件名为 showselect1.php，如代码 4-14 所示。

【代码 4-13】　select1.htm

```
…省略部分代码
        <form method=post action="showselect1.php">
        <select name="bmonth[]" multiple size=12>
        <option value="01" selected>1</option>
        <option value="02">2</option>
        <option value="03">3</option>
        <option value="04">4</option>
        <option value="05">5</option>
        <option value="06">6</option>
        <option value="07">7</option>
        <option value="08">8</option>
        <option value="09">9</option>
        <option value="10">10</option>
        <option value="11">11</option>
        <option value="12">12</option>
         </select>
…省略部分代码
```

其中“size=12”表示的是列表框的高度为 12，这样方便用户选择。select1.htm 的执行结果如图 4-11 所示（可以按住 CTRL 键在图中进行多选）。

细心的读者会发现，在定义多选列表框的时候，将其名称定义为了数组的形式即“bmonth[]”，这是因为 PHP 在进行多选列表框数据处理的时候，将被选中的数据以数组的形式保存。如果名称只是简单的“bmonth”，虽然用户能够进行多选，但是提交到服务器之后，实际得到的将是最后选择的值，其他值并没有被 PHP 捕捉到。

【代码 4-14】 showselect1.php

```
<?php
echo "你选择的月份是：<BR>";
if (!empty($_POST['bmonth']))
foreach ($_POST['bmonth'] as $Mon)
    echo($Mon . "月<br>");
?>
```

在图 4-11 上单击“提交”按钮，将会提交到 showselect1.php 文件进行处理，处理的结果如图 4-12 所示。

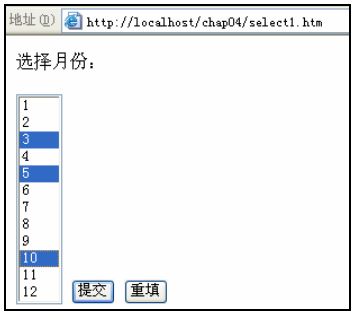


图 4-11 select1.htm 的执行结果

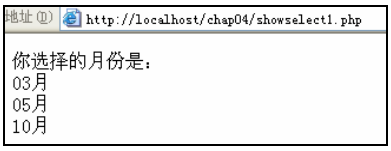


图 4-12 showselect1.php 的执行结果

4.2.5 隐藏字段

隐藏字段是允许用户把辅助信息附加到窗体上的完全不可见的控件。也就是说，隐藏字段将出现在浏览器的窗口中，但是用户无法修改，而且也不会受这些隐含字段的影响。通常这些字段是不可见的。

隐藏字段的表单控件的 HTML 语法结构如下：

```
<input type="hidden" name="age" value="20">
```

其中 type="hidden"指定了该控件是隐藏控件，后面的 name 和 value 属性同其他控件的定义是一样的。

隐藏字段通常用在页面跟踪的过程中。比如有时候需要将一个用户的姓名、密码、年龄等信息传到下一页，但是又不想这些信息被显示出来，此时可以使用隐含字段来实现。

实例 4-8 获取隐藏字段值

本实例演示如何获取隐藏字段的值。由两个文件组成，前台文件名为 hide.php，如代码 4-15 所示；后台文件名为 showhide.php，如代码 4-16 所示。

【代码 4-15】 hide.php

```
<html>
<head>
<title>
隐藏字段
</title>
</head>
<body>
```

```
<?php
    $username="dog";
?>
<form method=get action="showhide.php">
帐号：<INPUT maxLength=25 size=16 name=login ><br>
<input type="hidden" name="hidename" value="<?php print 'hello'; ?>">
<input type="submit" name="submit" value="提交">
<input type="reset" name="reset" value="重填">

</body>
<html>
```

提示　因为 hide.php 文件中使用了 php 代码所以文件名不能使用 hide.htm。

【代码 4-16】　showhide.php

```
<?php
    //获取隐藏字段信息
    echo "你的隐藏字段信息为：<br>";
    echo $_GET['hidename'] ;
?>
```

执行结果如图 4-13 所示。

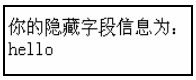


图 4-13　获取隐藏字段信息

实例 4-9　表单处理的综合应用

通过以上的介绍，相信读者已经了解了 PHP 中大致的表单信息处理过程。接下来编写本章开头介绍的代码 4-1 中的处理文件“showdetail.php”。该文件实现的功能就是把用户提交的数据输出出来。如果学习了数据库之后，就可以方便地将数据保存到数据库中。

【代码 4-17】　showdetail.php

```
<?php

    echo("你的帐号是：" . $_POST['login']);  //输出帐号
    echo("<br>");
    echo("你的姓名是：" .$_POST['yourname'] );  //输出姓名
    echo( "<br>");

    echo("你的密码是：" . $_POST['passwd']  );  //输出密码
    echo("<br>");

    echo("你的查询密码问题是：" . $_POST['question']  );  //查询密码问题
    echo("<br>");

    echo("你的查询密码答案是：" . $_POST['question2']   );  //查询密码答案
    echo("<br>");
```

```
        echo("你的出生日期是：" . $_POST['byear'] ."年". $_POST['bmonth'] . "月" .
$_POST['bday'] . "日"   );  //

    出生日期
        echo("<br>");
        echo("你的性别是：" . $_POST['gender']);  //性别
        echo("<br>");
        echo("你的爱好是：<br>" );  //爱好
        foreach ($_POST['hobby'] as $hobby)
            echo($hobby . "<br>");
    ?>
```

执行结果如图 4-14 所示。

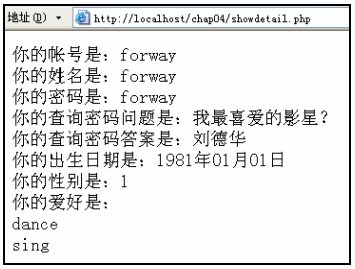


图 4-14　showdetail.php 的执行结果

4.3　常用表单数据验证

在用户提交表单数据的时候，首先要对提交的数据的合法性进行判断，如电话号码、用户名长度、电子邮件地址、身份证号码等。在进行判断的时候，通常采取的方法有两种，一种是客户端验证，另外一种是服务器端验证。

服务器端验证一般来说更安全，更具有隐蔽性，尤其需要同数据库打交道的时候只能通过服务器端验证。而客户端验证主要依靠客户端脚本，如 JavaScript 或者 VBScript 进行。

本节主要介绍如何在服务器端进行常用数据的验证。关于客户端脚本验证，将会在后面的“注册模块”进行讲解。

4.3.1　姓名验证

姓名验证是注册表单验证中最长用的。因为注册表单中的姓名会被保存进数据库中，而数据库表的设计一般都会对长度有要求。

实例 4-10　验证姓名

假如要限定姓名的长度在 5～50 之间，则可以用代码 4-18 来实现，文件名为 name.php。

【代码 4-18】　name.php

```
    <?php
    $name="123";
    if(empty($name))  //判断是否为空
    {
```

```
    die("姓名不能为空");
}
//判断长度
elseif ( (strlen($name) < 5) || (strlen($name) > 50))
{
    die("姓名长度必须在 5 和 15 之间");
}
Else   //如果合法，则输出姓名
{
    echo $name;
}
?>
```

以上的验证方式也适合一般的文本验证，如地址等信息。

name.php 的执行结果如图 4-15 所示。

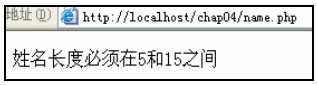


图 4-15　name.php 的执行结果

由于事先定义了“$name="123"”，$name 的长度只有 3，因此提示长度不够。

提示　die 的作用类似与 exit 函数，即输出内容后结束当前的 php 代码执行。

4.3.2　电子邮件验证

相比较上面的姓名验证，电子邮件验证稍微复杂一些。电子邮件最主要的特征是其中必须包含“@”符号，同时也可以进行长度限制。一般来说“@”符号后面，是一个邮件服务器地址，如果要求比较高的验证，可以进行 DNS 检测，判断是否真的有该邮件地址，实际中，一般不需要这么严格。

实例 4-11　电子邮件验证

电子邮件代码验证如代码 4-19 所示，文件名为 email.php。

【代码 4-19】　email.php

```
<?php
$email="hello 163.com";
if(empty($email))
{
    die("邮件地址不可以为空! ");
}
elseif ( (strlen($email) < 5) || (strlen($email) > 50))
{
    die("邮件地址长度必须在 5 和 50 之间");
}
elseif(!ereg("@",$email)) //必须包含@字符
{
    die("电子邮件不合法，请重输!");
}
```

```
else  //输出电子邮件
{
    echo $email;
}
?>
```

email.php 的执行结果如图 4-16 所示。



图 4-16　email.php 的执行结果

由于事先定义了“$email="hello 163.com"”，但$email 中不包含@字符，所以提示电子邮件地址不合法。

4.3.3　密码字段验证

密码验证的特点就是要限制密码的长度，然后需要确认两次输入的密码是否一致。为了保证两次密码的完全相同，需要验证两次密码的值以及数值类型是否一样。

实例 4-12　密码验证

密码验证如代码 4-20 所示，文件名为 password.php。

【代码 4-20】　password.php

```
<?php
$password="forway";
$cpassword="fprway" ;
if(empty($password) || empty($cpassword))
{
    die("密码不可空!");
}
elseif ( ((strlen($password) < 5) || (strlen($password) > 15)))
{
    die("密码长度在 5 和 15 之间");
}
//--- 值比较
elseif ( !(strlen($password) == strlen($cpassword)) )
{
    die("两次输入密码不匹配! ");
}
elseif( !($password === $cpassword)) //值和数据类型比较
{
    die("两次密码不匹配! ");
}
else  //循环输出密码，因为是密码所以输出*号
{
    for ($i=0;$i<strlen($password);$i++)
    {
        echo "*";
    }
```

```
}
?>
```

4.3.4　生日字段验证

生日字段也是常用的必须经过验证的字段。一般情况下，客户端的表单并不是做成文本框的形式让用户输入，而是使用下拉列表框的形式让用户进行选择。因此很多时候不需要进行复杂的验证。本实例将生日分成年、月、日 3 个部分来进行验证。即保证出生年在 1900～2005 之间，出生月在 1～12 之间，日在 1～31 之间。当然还可以进行更复杂的验证，即判断某个月份是否包含 31 号或者 30 号，不过很多时候没有这个必要。

实例 4-13　生日验证

生日字段验证如代码 4-21 所示，文件名为 birth.php。

【代码 4-21】　birth.php

```
<?php
$birth_month=1;
$birth_day=29;
$birth_year=1995;
if (empty($birth_month) || empty($birth_day) || empty($birth_year) )
{
    die("生日不完整!");
}
switch($birth_month)
{
    case 1: print "January "; break;
    case 2: print "February "; break;
    case 3: print "March ";break;
    case 4: print "April ";break;
    case 5: print "May "; break;
    case 6: print "June "; break;
    case 7: print "July "; break;
    case 8: print "August "; break;
    case 9: print "September "; break;
    case 10: print "October "; break;
    case 11: print "November "; break;
    case 12: print "December "; break;

    default: die("月份不合法!");
}
if (($birth_day < 1) || ($birth_day > 31))
{
    die("日期不对!!");
}
else
{
    echo $birth_day, " ";
}
if (($birth_year < 1900) || ($birth_year >2005))
```

```
    {
        die("年份不对!");
    }
    else
    {
        echo $birth_year;
    }
    ?>
```

birth.php 的输出结果如图 4-17 所示。

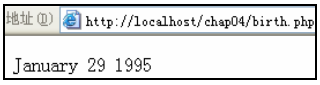


图 4-17　birth.php 执行结果

由于定义了月份为 1、日期为 29、年份为 1995，都符合设定的条件，结果输出完整的生日。如果将日期改成 32，则会输出“日期不对”的提示。

有关常用的字段服务器端验证就介绍到这里。在后面章节将会介绍客户端 JavaScript 的各种常用字段验证。

4.4　URL 编码解码函数

在前面介绍了 POST 和 GET 的基本知识，下面介绍一些常用的 URL 编码和解码的函数。

4.4.1　urlencode 函数

该函数将传入的字符串参数进行 URL 编码。该函数声明如下：

```
string urlencode (string str)
```

返回字符串中除了“-_.”之外的所有非字母数字字符都将被替换成百分号（%）后跟两位十六进制数，空格则编码为加号（+）。此编码与 WWW 表单 POST 数据的编码方式是一样的，同时与 application/x-www-form-urlencoded 的媒体类型编码方式一样。由于历史原因，此编码将空格编码为加号（+）方面与 RFC1738 编码（参见 rawurlencode()）不同。此函数便于将字符串编码并将其用于 URL 的请求部分，同时它还便于将变量传递给下一页。

实例 4-14　使用 urlencode 对 URL 进行编码

本实例演示了如何通过 urlencode 函数对 URL 进行编码，如代码 4-22 所示。

【代码 4-22】　urlencode.php

```
<?php
echo '<a href="mycgi?foo=', urlencode("张三"), '">张三</a>';
?>
```

执行结果如图 4-18 所示。



图 4-18　urlencode 函数应用

此时用户看不到编码的效果，但是查看源代码，会发现已经对关键字“张三”进行编码了，如下所示：

```
<a href="mycgi?foo=%D5%C5%C8%FD">张三</a>
```

4.4.2　urldecode 函数

该函数解码已编码的 URL 字符串。函数声明如下：

```
string urldecode(string str)
```

解码给出的已编码字符串中的任何%##。返回解码后的字符串。

实例 4-15　使用 urldecode 对 URL 进行解码

【代码 4-23】　urldecode.php

```
<?php
//要配合提交页面一起使用
$a = explode('&', $QUERY_STRING);
$i = 0;
while ($i < count($a)) {
   $b = split('=', $a[$i]);
   echo 'Value for parameter ', htmlspecialchars(urldecode($b[0])),
       ' is ', htmlspecialchars(urldecode($b[1])), "<br />\n";
   $i++;
}
?>
```

提示　还有两个进行 URL 编码和解码的函数，函数名分别为 rawurlencode()和 rawurldecode()函数，这两个函数是按照 RFC 1738 协议来进行的。有兴趣的读者可以参考 RFC 1738。

第二篇

高级技术篇

- PHP 中的文件处理技术
- PHP 与面向对象程序设计
- 错误调试与异常处理
- MySQL 数据库
- PHP 的数据库编程
- PHP 中的图像处理技术
- 会话控制

第 5 章　PHP 中的文件处理技术

文件处理技术是一种编程语言的基本处理能力之一。本章将介绍 PHP 中的各种文件处理技术和技巧。虽然很多数据都是保存在数据库中，但是文件管理依然占据着重要的地位。

文件处理一般包含以下一些步骤。

（1）打开文件。打开指定的文件，如果该文件不存在，则创建该文件。

（2）文件操作。在打开文件之后，就可以对该文件进行各种操作。如果文件中已经包含数据，则可以从中读取数据，重新向文件写入数据，或者向文件中添加数据。

（3）关闭文件。在完成处理文件之后，需要关闭该文件。

PHP 中提供了许多与文件相关的函数，可以用来将网页上的数据保存到文件，将文件中的数据检索到网页，或者仅仅修改已有的文件数据，或者添加新数据。其中最常用的文件操作如下：

（1）检查文件是否存在；

（2）打开文件；

（3）读取文件；

（4）向文件中写入数据。

下面详细介绍以上的操作。

5.1　检测文件或者目录是否存在

在 PHP 中，可以通过函数 file_exists 来检测一个文件或者目录是否存在。该函数声明如下：

```
bool file_exists ( string filename)
```

如果由 filename 指定的文件或目录存在则返回 TRUE，否则返回 FALSE。



在 Windows 中要用//computername/share/filename 或者\\computername\share\filename 来检查网络中的共享文件。

实例 5-1　检测文件是否存在

本实例使用 file_exists 函数检测文件是否存在，如代码 5-1 所示。

【代码 5-1】　file_exists.php

```
<?php
$filename = 'hello.txt';
$catalog="hello";

//文件检测
```

```
    if (file_exists($filename)) {
        print $filename."文件存在!<br>";
    } else {
        print $filename."文件不存在!<br>";
    }

    //目录检测
    if (file_exists($catalog)) {
        print $catalog."目录存在!";
    } else {
        print $catalog."目录不存在!";
    }
    ?>
```

file_exists.php 文件的执行结果如图 5-1 所示。



图 5-1　file_exists.php 文件的执行结果

由于笔者已在当前目录下创建了目录“hello”，所以执行结果显示该目录存在，而当前目录下没有“hello.txt”文件，因此提示文件不存在。

5.2　打 开 文 件

任何对文件的操作都必须先要打开该文件。

在 PHP 中，打开一个文件使用的函数是 fopen。

fopen 函数的声明如下所示：

```
    int fopen ( string filename, string mode [, int use_include_path [, resource
zcontext]])
```

调用 fopen 函数的时候，需要传递 2 个、3 个或者 4 个参数。通常使用两个参数。其中第一个参数 filename 表示要打开的文件名，而第二个参数 mode 表示打开的方式。不同的打开方式，可以进行的操作是不一样的。

fopen()将 filename 指定的名字资源绑定到一个流上。如果 filename 是“scheme://...”的格式，则被当成一个 URL，PHP 将搜索协议处理器（也被称为封装协议）来处理此模式。如果该协议尚未注册封装协议，PHP 将发出一条消息来帮助检查脚本中潜在的问题并将 filename 当成一个普通的文件名继续执行下去。

如果 PHP 认为 filename 指定的是一个本地文件，将尝试在该文件上打开一个流。该文件必须是 PHP 可以访问的，因此需要确认文件访问权限是否允许该访问。如果激活了安全模式或者 open_basedir 则会应用进一步的限制。

如果 PHP 认为 filename 指定的是一个已注册的协议，而该协议被注册为一个网络 URL，PHP 将检查并确认 allow_url_fopen 已被激活。如果关闭了，PHP 将发出一个警告，而 fopen 的调用则失败。

可以这样理解，如果参数 filename 以 http://开头，则表示打开的是远程 Web 服务器上的文件，如果以 ftp://开头，则表示要从远程 FTP 服务器上获取文件，并需要与指定的服务器建

立 FTP 会话。如果前面没有任何前缀，一般情况下表示打开本地文件。

第 2 个参数 mode 有多种取值，用来指示文件是读、写还是附加。mode 参数的具体取值如表 5-1 所示。

表 5-1 fopen 函数中 mode 的可能值列表

mode	说　明
'r'	只读方式打开，将文件指针指向文件头
'r+'	读写方式打开，将文件指针指向文件头
'w'	写入方式打开，将文件指针指向文件头并将文件大小设为零。如果文件不存在则尝试创建
'w+'	读写方式打开，将文件指针指向文件头并将文件大小设为零。如果文件不存在则尝试创建
'a'	写入方式打开，将文件指针指向文件末尾。如果文件不存在则尝试创建
'a+'	读写方式打开，将文件指针指向文件末尾。如果文件不存在则尝试创建
'x'	创建并以写入方式打开，将文件指针指向文件头。如果文件已存在，则 fopen()调用失败并返回 FALSE，生成一条 E_WARNING 级别的错误信息。如果文件不存在则尝试创建。这和给底层的 open(2)系统调用指定 O_EXCL\|O_CREAT 标记是等价的。此选项仅能用于本地文件
'x+'	创建并以读写方式打开，将文件指针指向文件头。如果文件已存在，则 fopen()调用失败并返回 FALSE，并生成一条 E_WARNING 级别的错误信息。如果文件不存在则尝试创建。这和给底层的 open(2)系统调用指定 O_EXCL\|O_CREAT 标记是等价的。此选项仅能用于本地文件
'b'	二进制模式。用于与其他模式进行连接。如果文件系统能够区分二进制文件和文本文件，可能会使用它。Windows 系统可以区分；而 Unix 系统则不区分。推荐一直使用这个选项，以便获取最大程度的可移植性。二进制模式是默认的模式
't'	用于与其他模式的结合。这个模式只是 Windows 系统下的一个选项。不是推荐选项，除非曾经在代码中使用了 b 选项



如果指定的文件需要从远程的 HTTP 服务器上获取，则只能以只读方式打开。这样可以有效地防止他人篡改网页。类似地，如果需要从 FTP 服务器上获取指定的文件，则只能以写或者读的方式打开，但不能同时以写和读的方式打开。

在一些情况下，可能会遇到方式 b，这种方式只适合一定的平台，如 windows，它能区分文本和二进制格式。这种方式在诸如 UNIX 的平台上不起作用。



不同的操作系统产品具有不同的行结束习惯。当写入一个文本文件并想插入一个新行时，需要使用符合操作系统的行结束符号。基于 UNIX 的系统使用\n 作为行结束字符，基于 Windows 的系统使用\r\n 作为行结束字符，基于 Macintosh 的系统使用\r 作为行结束字符。如果写入文件时使用了错误的行结束符号，则其他应用程序打开这些文件时可能导致数据发生改变，不是原来写入的数据。

fopen 的第 3 个参数 use_include_path 是可选的，在按照由该参数指定的路径下查找文件时使用它。如果希望在 php.ini 文件中设定的 include_path 路径中进行查找，则只需要将该参数设置为 1。

常用的打开文件的方法如代码 5-2 所示。

【代码 5-2】

```
<?php
//只读方式打开文本文件，该文件路径为相对当前 PHP 文件的路径
$handle = fopen ("/home/rasmus/file.txt", "r");
```

```
//只写方式打开二进制文件，该文件路径为相对当前 PHP 文件的路径
$handle = fopen ("/home/rasmus/file.gif", "wb");
//打开远程 WEB 服务器文件
$handle = fopen ("http://www.example.com/", "r");
//打开远程 FTP 文件
$handle = fopen ("ftp://user:password@example.com/somefile.txt", "w");
//打开本地机器上的绝对路径的文件，该种方式适合 Unix 系统和 Windows 系统
$handle = fopen ("c:/data/info.txt", "r");
//打开本地机器上的绝对路径的文件，该种方式适 Windows 系统，如果使用反斜线，必须使用转义符
//即双反斜线
$handle = fopen ("c:\\data\\info.txt", "r");
?>
```

如果用户以“w”或者“a”模式打开一个文件时，该文件不存在，则系统会自动创建该文件。

实例 5-2　创建文件

本实例通过 fopen 函数创建一个文件，如代码 5-3 所示。

【代码 5-3】　createfile.php

```
<?php
//在 hello 目录下创建一个 1.txt 文件，以只写模式创建
$handle = fopen ("hello/1.txt", "w");
//关闭文件，使用 fclose 函数
fclose($handle);
echo "/hello/1.txt 文件创建成功!<br>";
//在 C 盘根目录下创建一个 temp.txt 文件，以追加模式创建
$handle=fopen("c:/temp.txt","a");
//同样也要关闭文件
fclose($handle);
echo "c:/temp.txt 文件创建成功!";
?>
```

createfile.php 文件的执行结果如图 5-2 所示。



图 5-2　createfile.php 文件的执行结果

执行代码 5-3 后，可以去 C 盘根目录下查看刚才创建的 temp.txt 文件，同时也可以在当前目录的子目录“hello”下创建了 1.txt 文件。注意，要确保“hello”目录已经创建。

以上代码中虽然创建了文件，但是整个文件是空的，因此接下来的步骤是向文件中添加数据。

5.3　写入或者追加数据

5.3.1　写入数据

在 PHP 中，写文件的函数有好几个，最常用的函数为 fwrite()，也可以使用 fputs()函数，该函数是 fwrite()函数的别名函数（两者的用法一样）。

fwrite()函数的声明如下：

```
int fwrite ( resource handle, string string [, int length])
```

fwrite()把 string 的内容写入文件指针 handle 处。如果指定了 length，当写入了 length 个字节或者写完了 string 以后，写入就会停止。

fwrite()返回写入的字符数，出现错误时则返回 FALSE。


注意 如果给出了 length 参数，则 magic_quotes_runtime 配置选项将被忽略，而 string 中的斜线将不会被去掉。

实例 5-3 将数据写入文件

本实例以只写方式打开一个文件，然后向文件中写入一行文字，如代码 5-4 所示。

【代码 5-4】 writedata.php

```
<?php
$filename = 'hello/1.txt';
$somecontent = "我先被写入的!\r\n";
$somecontent1 = "我后被写入的!\r\n";

// 首先要确定文件存在并且可写
if (is_writable($filename)) {

    // 在这个例子里，将使用只写模式打开$filename
    // 文件指针将会在文件的开头,如果原来文件中有其他数据，则会被删除

    if (!$handle = fopen ($filename, "w")) {
         print "不能打开文件 $filename";
         exit;
    }
    // 将$somecontent 写入到打开的文件中
    if (!fwrite($handle, $somecontent)) {
        print "不能写入到文件 $filename";
        exit;
    }
    print "成功地将\" $somecontent \"写入到文件$filename<br>";
    fclose($handle);
     $handle = fopen ($filename, "w");  //再次打开
     //继续打开文件写入另外一个字符串，原来写入的数据则被清除
    fwrite($handle, $somecontent1);
    fclose($handle);
     print "成功地将\" $somecontent1 \"写入到文件$filename";
} else {
    print "文件 $filename 不可写";
}
?>
```

writedata.php 文件的执行结果如图 5-3 所示。

从图 5-3 中可以看出，虽然两句话都被写入到文件中，而实际上只有最后一句话被写入到文件了，如图 5-4 所示。

这是因为采用了只写模式打开，每次写入新数据的时候，原始数据都会被删除。

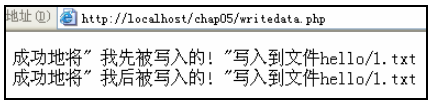


图 5-3　writedata.php 的执行结果

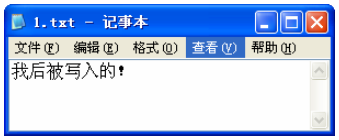


图 5-4　最终被写入的数据

实例 5-4　写入多行数据

本实例实现一次将多行数据写入文件，如代码 5-5 所示。

【代码 5-5】　writedata1.php

```
<?php
$filename = 'hello/2.txt';
$handle = fopen ($filename, "w");
fwrite($handle, "1\r\n");
fwrite($handle, "2\r\n");
fwrite($handle, "3\r\n");
fwrite($handle, "4\r\n");
fwrite($handle, "5\r\n");
fwrite($handle, "6\r\n");
fwrite($handle, "7\r\n");
fwrite($handle, "8\r\n");
fclose($handle);
?>
```

执行代码 5-5 后，会创建 2.txt 文件，2.txt 文件的内容如图 5-5 所示。

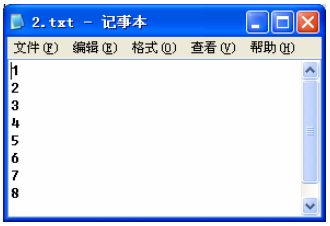


图 5-5　写入多行文本

从图 5-5 中可以看出，打开一个文件后可以一次写入多行文本。

提示　\r\n 在 Windows 中表示字符串结束并且换行。

5.3.2　追加数据

前面介绍了向文件中写入数据，但是每次重新打开文件后，都会删除原来的数据，这样很不方便。比如要将一些个人的基本信息保存在一个文本文件中，但不是一次性写入，而是需要经常打开文件追加，这个时候，就需要使用追加模式来写入数据。

实例 5-5　追加数据

本实例实现打开一个已有文件，然后追加数据，如代码 5-6 所示。

【代码 5-6】 appenddata.php

```
<?php
$filename = 'hello/student.txt';
$student1 = "姓名:小狗 \t 年龄:5 \t 性别:男\r\n";
$student2 = "姓名:小猫 \t 年龄:4 \t 性别:女\r\n";
$student3 = "姓名:小猪 \t 年龄:3 \t 性别:男\r\n";
   // 在这个例子里，首先使用只写模式模式打开$filename

    if (!$handle = fopen ($filename, "w")) {
        print "不能打开文件 $filename";
        exit;
    }
    // 将$student1 写入到打开的文件中
    if (!fwrite($handle, $student1)) {
       print "不能写入到文件 $filename";
       exit;
    }
    //fwrite($handle, $student1);
    print "成功地将\" $student1 \"写入到文件 $filename<br>";
    fclose($handle);
    $handle = fopen ($filename, "a");  //接着以添加模式打开文件
    //继续添加其他信息
    fwrite($handle, $student2);
    print "成功地将\" $student2 \"写入到文件 $filename<br>";
    fwrite($handle, $student3);

    print "成功地将\" $student3 \"写入到文件 $filename<br>";
    fclose($handle);
?>
```

appenddata.php 文件的执行结果如图 5-6 所示。

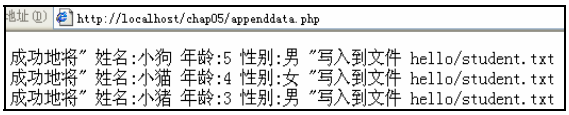


图 5-6 appenddata.php 文件的执行结果

打开刚才创建的 student.txt，可以看到如图 5-7 所示结果。

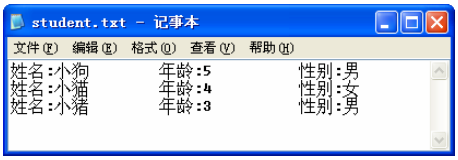


图 5-7 添加数据成功

5.4 关 闭 文 件

任何文件在打开操作后都必须关闭，否则可能会引起错误。在前面的例题中，其实已经使用了关闭文件函数，即 fclose()函数。fclose()函数的声明如下：

```
bool fclose ( resource handle)
```

fclose 函数将 handle 指向的文件关闭。如果成功则返回 TRUE，失败则返回 FALSE。

提示　文件指针必须有效，并且是通过 fopen()或 fsockopen()成功打开的。

简单示例如代码 5-7 所示。

【代码 5-7】

```
<?php
    $handle = fopen('somefile.txt', 'r');
    fclose($handle);
?>
```

5.5　读 取 数 据

前面介绍了如何创建文件、打开文件以及向文件中写入数据以及追加数据，接下来将介绍如何从文件中读取数据。

5.5.1　读取一行数据

打开文件后，如果要逐行读取文件内容，可以使用下面的函数。

（1）ftets 函数。该函数的声明如下：

```
string fgets(int handle [, int length])
```

其中 handle 参数是被打开的要读取数据的句柄，而 length 参数表示要读取的数据长度。该函数的作用是从 handle 指向的文件中读取一行并返回长度最多为 length–1 字节的字符串。遇到换行符（包括在返回值中）、EOF 或者已经读取了 length–1 字节后停止。如果没有指定 length，则默认为 1KB，或者说 1024 字节。出错时返回 FALSE。

实例 5-6　一次读入一行数据

本实例演示了 fgets 函数的应用，如代码 5-8 所示。

【代码 5-8】　fgets.php

```
<?php
$handle = fopen ("hello/student.txt", "r");
while (!feof ($handle)) {
   $buffer = fgets($handle,1024);
   echo $buffer . "<br>";
}
fclose ($handle);
?>
```

fgets.php 文件的执行结果如图 5-8 所示。

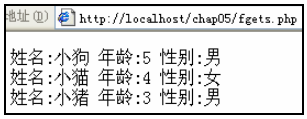


图 5-8　fgets.php 的执行结果

实例中使用代码循环读取了前面创建的 student.txt 文件中的数据。

提示 feof 函数为判断是否到了文件最后的函数，在后面会有详细介绍。

（2）fgetss 函数。fgetss 函数是 fgets 函数的一个变体。其功能同 fgets 函数基本类似，只是 fgetss 函数会过滤掉被读取内容中的 HTML 和 PHP 标记。

fgetss 函数的声明如下：

```
string fgetss( resource handle, int length [, string allowable_tags])
```

fgetss 函数和 fgets()相同，只除了 fgetss 尝试从读取的文本中去掉任何 HTML 和 PHP 标记。可以用可选的第三个参数指定哪些标记不被去掉。当读取其他人写的文件的时候，可以设置过滤一些恶意代码。因为一些恶意的 HTML 和 PHP 代码，可以很容易控制服务器。

实例 5-7 读取数据并过滤 HTML 和 PHP 标记

本实例演示了 fgetss 函数的应用，如代码 5-9 所示。

【代码 5-9】 fgetss.php

```
<?php
$filename = 'hello/3.txt';
$handle = fopen ($filename, "w");
fwrite($handle, "<b>i like dancing very much!</b>\r\n");
fwrite($handle, "<br>i like singing very much!\r\n");
fclose($handle);
echo "fgets 函数的输出结果：<br>";
$handle = fopen ("hello/3.txt", "r");
while (!feof ($handle)) {
    $buffer = fgets($handle,1024);  //fgets 函数，不过滤<b>、<br>等 html 标记
    echo $buffer;
}
fclose ($handle);
echo "<br><br>fgetss 函数的输出结果：<br>";
$handle = fopen ("hello/3.txt", "r");
while (!feof ($handle)) {
    $buffer = fgetss($handle,1024);  //fgetss 函数，过滤<b>、<br>等 html 标记
    echo $buffer;
}
fclose ($handle);
?>
```

fgetss.php 文件的执行结果如图 5-9 所示。

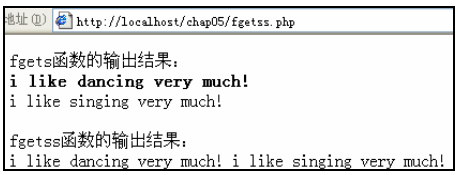


图 5-9 fgetss.php 文件的执行结果

从图 5-9 中可以看出，由于 fgets 函数没有过滤，因此，在显示的时候将加粗显

示字体，而
标记将会换行。而用 fgetss 函数则过滤了这几个 html 标记，所以在浏览器中的显示，没有将字体加粗，同时也没有换行。读者也可以右键单击浏览器，在弹出菜单中选择【查看源文件】，可以看到两次输出的内容分别为：

```
//fgets 函数输出
<b>i like dancing very much!</b>
<br>i like singing very much!

//fgetss 函数输出
i like dancing very much!
i like singing very much!
```

（3）fgetcsv()函数。fgetcsv 函数也是 fgets 函数的一个变体。该函数从文件指针中读入一行并解析 CVS 字段，该函数的声明如下：

```
array fgetcsv ( int handle, int length [, string delimiter [, string enclosure]])
```

和 fgets()类似，只除了 fgetcsv()解析读入的行并找出 CVS 格式的字段然后返回一个包含这些字段的数组。可选的第三个参数 delimiter 的默认值是逗号。可选参数 enclosure 的默认值是双引号。delimiter 和 enclosure 都被限制为一个字符。如果多于一个字符，则只使用第一个字符。length 必须大于 CVS 文件中长度最大的行（以便于处理行结束字符）。fgetcsv()出错时返回 FALSE，包括碰到文件结束时。



CVS 文件中的空行将返回为一个包含有单个 null 字段的数组，而不会被当成错误。

如果在文件中使用了分隔符号，如逗号、制表符等，可以使用 fgetcsv 函数将文件中读入的一行分成多行。比如在 appenddata.php 中，向 student.txt 文件中写入了“$student1 = "姓名:小狗\t 年龄:5\t 性别:男\r\n"”的记录，如果现在想将姓名、年龄以及性别分成三行显示，此时可以通过分隔符“\t”来将该行分解成 3 个字段，并保存在一个数组中。

实例 5-8　使用 fgetcsv 函数解析 CVS 文件

本实例介绍 fgetcsv 函数的应用，代码 5-10 所示。

【代码 5-10】　fgetcsv.php

```
<?php
$row = 1;
$handle = fopen ("hello/student.txt","r");
while ($data = fgetcsv ($handle, 1000, "\t")) {
    $num = count ($data);  //统计数组大小
    print "<p> 在第 $row 行中有 $num 个字段: <br>\n";
    $row++;
    for ($c=0; $c < $num; $c++) {
        print $data[$c] . "<br>\n";
    }
}
fclose ($handle);
?>
```

fgetcsv.php 文件的执行结果如图 5-10 所示。

从图 5-10 中可以看出，文件 student.txt 中的每行文本，都通过分隔符“\t”分隔开，然后分别输出。



图 5-10　fgetcsv.php 文件的执行结果

5.5.2　读取整个文件

前面介绍的 3 个函数主要用于逐行读取文件中的数据，但是很多时候不需要逐行读取数据，而是要一次读取整个文件的数据。PHP 中提供了 4 个函数用于读取整个文件的数据。

（1）readfile()函数。readfile 函数的声明如下：

```
int readfile ( string filename [, bool use_include_path [, resource context]])
```

readfile 函数读入一个文件并写入到输出缓冲。该函数返回从文件中读入的字节数。如果出错返回 FALSE，并且除非是以@readfile()形式调用，否则会显示错误信息。

该函数首先打开目标文件，并且将文件的内容输出到标准输出，如浏览器中，然后关闭该文件。使用该函数比较简单，不需要打开和关闭文件。

实例 5-9　读取整个文件

本实例使用 readfile 函数读取整个文件的内容，如代码 5-11 所示。

【代码 5-11】　readfile.php

```
<?php
readfile("hello/student.txt");
?>
```

代码 5-11 比较简单，直接读取 student.txt 文件中的数据，并显示在浏览器中，运行结果如图 5-11 所示。

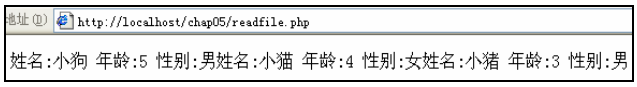


图 5-11　readfile.php 文件执行结果

（2）fpassthru()函数。fpassthru 函数输出文件指针处的所有剩余数据，该函数的声明如下：

```
int fpassthru (resource handle)
```

该函数将给定的文件指针从当前的位置读取到 EOF 并把结果写到输出缓冲区。如果发生错误，fpassthru()返回 FALSE。否则 fpassthru()返回从 handle 读取并传递到输出的字符数目。

如果既不修改文件也不在特定位置检索，只想将文件的内容下载到输出缓冲区，应该使用 readfile()，这样可以省去 fopen()调用。

提示　当在 Windows 系统中将 fpassthru()用于二进制文件时，要确保在用 fopen()打开文件时在 mode 中附加了“b”选项来将文件以二进制方式打开。建议在处理二进制文件时使用“b”标志，这样可以使脚本的移植性更好。

同 readfile 相比，fpassthru 需要首先打开文件，数据读取完毕还需要关闭文件。另外 fpassthru 函数是读取当前指针之后的所有数据。假如当前指针已经在文件的中间，则只是输出所有后面的数据。

实例 5-10　读取当前指针后的全部文件内容

本实例演示了如何通过 fpassthru 函数读取当前指针后的全部内容，如代码 5-12 所示。

【代码 5-12】　fpassthru.php

```
<?php
$filename = 'hello/student.txt';
$handle = fopen ($filename, "r");
$buffer = fgets ($handle,1024);  //首先读取一行，移动指针

//接着调用 fpassthru 输出剩下内容
fpassthru($handle);
fclose($handle);
?>
```

fpassthru.php 的执行结果如图 5-12 所示。

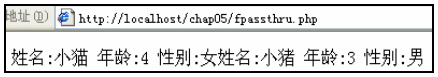


图 5-12　fpassthru.php 文件执行结果

从图 5-12 可以看出，由于在代码中首先调用了 fgets 函数读取了一行，因此接着调用 fpassthru 函数，只会输出接下来的两行内容（读者可以对比图 5-11 和图 5-12）。

实例 5-11　向浏览器发送图像

本实例用来向浏览器输出一幅图像，如代码 5-13 所示。

【代码 5-13】　fpassthru_pic.php

```
<?php
$filename = 'hello/1.bmp';
$handle = fopen ($filename, "rb");  //二进制读取数据，使用 rb 模式

//发送 html 头，表示发送二进制数据
header("Content-Type: image/png");
//filesize 为获取文件大小，在后面介绍
header("Content-Length: ".filesize($filename));

//输出图像数据到浏览器
fpassthru($handle);
exit;
?>
```

fpassthru_pic.php 文件的执行结果如图 5-13 所示。

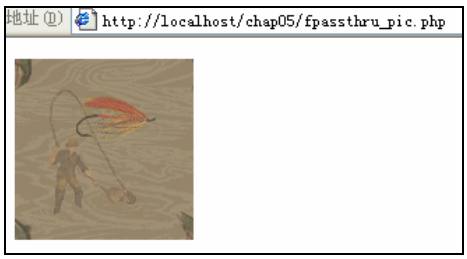


图 5-13　fpassthru_pic.php

（3）file()函数。file 函数也是读取整个文件的内容，但是该函数会把整个文件读入一个数组中。file 函数的声明如下：

```
array file ( string filename [, int use_include_path [, resource context]])
```

和 readfile()一样，只是 file()将文件作为一个数组返回。数组中的每个单元都是文件中相应的一行，包括换行符在内。如果失败，file()返回 FALSE。如果想在 include_path 中搜寻文件，可以将可选参数 use_include_path 设为“1”。

提示　返回的数组中每一行都包括了行结束符，因此，如果不需要行结束符，可以使用 trim()函数。

实例 5-12　将整个文件读入到数组

本实例使用 file 函数将整个文件读入到数组并输出，如代码 5-14 所示。

【代码 5-14】　file.php

```
<?php
// 将一个文件读入数组，这里读取的是远程文件
$lines = file ('http://www.sohu.com');
// 在数组中循环，显示 html 的源文件并加上行号
foreach ($lines as $line_num => $line) {
    echo "Line #<b>{$line_num}</b> : " . htmlspecialchars($line) . "<br>\n";
}
?>
```

file.php 文件的执行结果如图 5-14 所示。

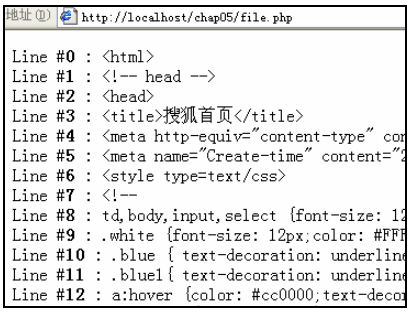


图 5-14　file.php 文件的执行结果

代码 5-14 通过函数 file 打开 sohu 网站的首页，然后将获取的内容按行编号输出。从代码中很容易理解，file 函数是将整个文件的内容按行保存在了数组中。

（4）file_get_contents 函数。file_get_contents 函数将整个文件作为一个字符串读入。该函数的声明如下：

```
string file_get_contents ( string filename [, int use_include_path [, resource context]])
```

file_get_contents()函数是将文件的内容读入到一个字符串中的首选方法。将在参数 offset 所指定的位置开始读取长度为 maxlen 的内容。如果失败，file_get_contents()将返回 FALSE。如果操作系统支持还会使用内存映射技术来增强性能。除了 file_get_contents()将文件返回为一个字符串以外，其余与 file()的使用一样。

提示　该函数适用于二进制对象。

实例 5-13　将整个文件读入到字符串

本实例使用 file_get_contents 函数将文件读入到字符串，如代码 5-15 所示。

【代码 5-15】　file_get_contents.php

```
<?php
// 将文件作为字符串读入到一个变量中
$lines = file_get_contents ('hello/student.txt');

//输出内容
echo $lines;
?>
```

file_get_contents.php 的执行结果如图 5-15 所示。

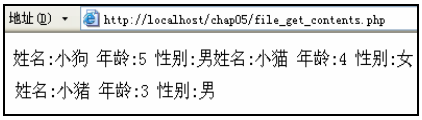


图 5-15　file_get_contents.php 文件的执行结果

5.5.3　读取一个字符

很多时候需要对整个文件的每个字符进行分析，如查找、替换部分内容等。PHP 也提供了相应的函数用来逐个读取数据。

（1）fgetc()函数。fgetc()函数从文件指针中读取字符，该函数的声明如下：

```
string fgetc (resource handle)
```

该函数返回包含有一个字符的字符串，该字符从 handle 指向的文件中得到。碰到文件结束即 EOF 为真，否则返回 FALSE。

实例 5-14　读入单个字符

本实例使用 fgetc 函数读入文件中的单个字符，如代码 5-16 所示。

【代码 5-16】　fgetc.php

```
<?php
$fp = fopen('hello/4.txt', 'r');
```

```
if (!$fp) {
    echo '不能打开文件!';
}
while (false !== ($char = fgetc($fp))) {
    echo "$char";
}

fclose($fp);
?>
```

fgetc.php 通过 fgetc 函数逐个读取文件 4.txt 中的字符。



注意 如果文件中包含有中文，则可能不能正确显示结果。因为一个中文字符包含两个字节，而 fgetc 通常只会读取一个字节的字符。

5.5.4 读取任意长度函数

另外一个读取文件的函数 fread()，能够从文件中读取指定长度的数据。该函数的声明如下：

```
string fread ( int handle, int length)
```

fread()从文件指针 handle 读取最多 length 个字节。该函数在读取完 length 个字节数，或到达 EOF 的时候，或（对于网络流）当一个包可用时就会停止读取文件。同时该函数也能够读取二进制文件。

实例 5-15 读入指定长度的字符串

本实例演示了 fread 函数的应用，如代码 5-17 所示。

【代码 5-17】 fread.php

```
<?php
$fp = fopen('hello/4.txt', 'r');
//读取文件的前 10 个字符
$contents = fread($fp, 10);
echo $contents."<br><br>";
fclose($fp);

//读取网络数据包数据
$handle = fopen ("http://www.sohu.com/", "rb");
$contents = "";
do {
    $data = fread($handle, 8192);
    if (strlen($data) == 0) {
        break;
    }
    $contents .= $data;
} while(true);

echo $contents;
fclose ($handle);
?>
```

fread.php 的运行结果如图 5-16 所示。

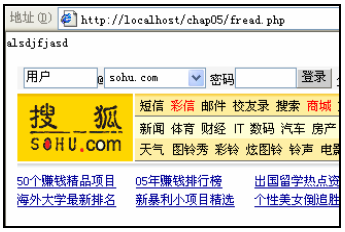


图 5-16　fread.php 文件的执行结果

代码 5-17 首先从 4.txt 文件读取了指定的 10 个字符，然后打开远程文件下载数据。这里打开的是搜狐的主页。

提示　上例比使用 while(!feof())的方法性能要好，因为在每个循环中节约了函数调用的花费。

5.5.5　文件定位

前面虽然介绍了逐行读取数据、逐个字符读取数据等，但是通过这些函数来定位文件指针效率比较低。PHP 中有专门进行文件定位的函数。

（1）rewind()函数。rewind 函数将文件位置指针设为文件流的开头，其声明如下：

```
bool rewind ( resource handle)
```

如果成功则返回 TRUE，失败则返回 FALSE。文件指针必须合法，并且指向由 fopen()成功打开的文件。

提示　如果将文件以附加（"a"）模式打开，写入文件的任何数据总是会被附加在后面，不管文件指针的位置。

实例 5-16　将文件指针指向文件头

本实例演示了 rewind 函数的应用，如代码 5-18 所示。

【代码 5-18】　rewind.php

```
<?php
$handle = fopen ("hello/student.txt", "r");
//首先读取第一行
$buffer = fgets($handle,1024);
echo $buffer . "<br>";
//读取第二行
$buffer = fgets($handle,1024);
echo $buffer . "<br>";
//将指针回到文件开始，继续读取一行数据，将读取到第一行数据
rewind($handle);
$buffer = fgets($handle,1024);
echo $buffer . "<br>";
fclose ($handle);
?>
```

rewind.php 的执行结果如图 5-17 所示。

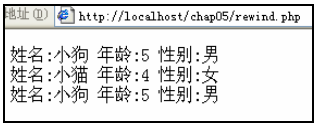


图 5-17　rewind.php 文件的执行结果

从图 5-18 可以看出，在使用 rewind 函数之后，指针又回到了文件开始，所以最后一行为文件的第一行数据。

（2）fseek()函数。fseek()函数用于文件指针定位，该函数的声明如下：

```
int fseek ( resource handle, int offset [, int whence])
```

fseek 函数在文件中设定文件指针位置。新指针的位置，从文件头开始以字节数度量，是在参数 whence 指定的位置加上 offset。

whence 的值定义如下。

SEEK_SET：设定位置等于 offset 字节。

SEEK_CUR：设定位置为当前位置加上 offset。

SEEK_END：设定位置为文件尾加上 offset（要移动到文件尾之前的位置，需要给 offset 传递一个负值）。

如果没有指定 whence，默认为 SEEK_SET。

如果函数调用成功则返回 0；否则返回–1。

注意　　移动到 EOF 之后的位置不算错误。

实例 5-17　设定文件指针位置

本实例演示了 fseek 函数的应用，如代码 5-19 所示。

【代码 5-19】　fseek.php

```
<?php
$handle = fopen ("hello/5.txt", "r");
//首先读取第一行，读取完毕后，指针在第一行最后
$buffer = fgets($handle,1024);
echo $buffer . "<br>";
//接着将指针向后移动 5 个字节，因为是相对当前位置，所以需要设定 SEEK_CUR 参数
fseek($handle,5,SEEK_CUR);
//接着读取一行
$buffer = fgets($handle,1024);
echo $buffer . "<br>";
//将指针回到文件开始，重新读取第一行，因为是绝对位置为 0，
//则不用设置 whence 参数默，默认的 whence 是 SEEK_SET，表示绝对位置
fseek($handle,0);
//读取一行数据，因为指针回到文件开始，所以会读取第一行数据
$buffer = fgets($handle,1024);
echo $buffer . "<br>";
fclose ($handle);
?>
```

fseek.php 文件的执行结果如图 5-18 所示。

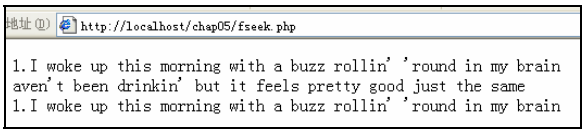


图 5-18　fseek.php 文件的执行结果

代码 5-19 打开了文件 5.txt，然后先读取第一行数据，接着将指针向后移动 5，读取第二行中第 5 个字符后面剩下的字符。然后将指针设置为文件开始，并获取一行数据。

5.txt 的文件内容如下所示：

```
1.I woke up this morning with a buzz rollin' 'round in my brain
2.I haven't been drinkin' but it feels pretty good just the same
3.It must be contagious-looks like it's goin' around
```

从上面可以看出，rewind 函数等于一个 0 偏移量的 fseek 函数。

（3）ftell()函数。ftell 函数并不起实际的定位作用，而是报告当前文件指针的位置。该函数的声明如下：

```
int ftell(resource handle)
```

返回由 handle 指定的文件指针的位置，即文件流中的偏移量。如果出错，返回 FALSE。

实例 5-18　获取文件指针位置

本实例演示了 ftell 函数的应用，如代码 5-20 所示。

【代码 5-20】　ftell.php

```
<?php
// 打开文件并读些数据
$fp = fopen("hello/5.txt", "r");
//获得前 15 个字符
$data = fgets($fp, 15);
// 获取当前指针
echo ftell($fp); // 输出当前位置
fclose($fp);
?>
```

代码 5-20 的输出结果为 14，因为之前已经获取了 15 个字符，而指针从 0 开始计算，所以当前位置为 14。

5.6　其他常用文件处理函数

前面介绍了最常用的文件处理的函数，接下来介绍其他的一些常用函数，这些函数能够帮助进行文件处理。

（1）filesize()函数。filesize 函数用于获取一个文件的大小。该函数声明如下：

```
int filesize ( string filename)
```

该函数返回文件大小的字节数，如果出错返回 FALSE。

提示　因为 PHP 的整数类型是有符号的，并且大多数平台使用 32 位整数，filesize()函数在碰到大于 2GB 的文件时可能会返回非预期的结果。对于 2 ~ 4GB 之间的文件，通常可以使用 sprintf("%u", filesize($file))来克服此问题。

（2）unlink()函数。unlink 函数用于删除一个文件。该函数的声明如下：

```
bool unlink (string filename)
```

如果成功则返回 TRUE，失败则返回 FALSE。

（3）rmdir()函数。rmdir 函数用于删除一个目录。该函数声明如下：

```
bool rmdir (string dirname)
```

该函数尝试删除 dirname 所指定的目录。该目录必须是空的，而且要有相应的权限。如果成功则返回 TRUE，失败则返回 FALSE。

（4）mkdir()函数。mkdir 函数用于创建一个目录。该函数的声明如下：

```
bool mkdir (string pathname [, int mode])
```

尝试新建一个由 pathname 指定的目录。mode 是指操作权限，默认的 mode 是 0777，表示最大可能的访问权。

（5）basename 函数。该函数返回路径中的文件名部分。声明如下：

```
string basename ( string path [, string suffix])
```

path 参数给出一个包含有指向一个文件的全路径的字符串，函数返回基本的文件名。如果文件名是以 suffix 结束的，则这一部分也会被去掉。

在 Windows 中，斜线（/）和反斜线（\）都可以用作路径分隔符。在其他环境下是斜线（/）。该函数的示例如下：

```
<?php
$path = "/home/httpd/html/index.php";
$file = basename ($path);       // $file is set to "index.php"
$file = basename ($path,".php"); // $file is set to "index"
?>
```

（6）dirname()函数。该函数返回路径中的目录部分。声明如下：

```
string dirname ( string path)
```

path 参数给出一个包含有指向一个文件的全路径的字符串，函数返回去掉文件名后的目录名。示例如下：

```
<?php
$path = "/etc/passwd";
$file = dirname ($path); // $file is set to "/etc"
?>
```



PHP 4.0.3 中，dirname()被修正为 POSIX 兼容。本质上就意味着如果在 path 中没有斜线，则返回一个点（'.'），表示当前目录。否则返回的是把 path 中结尾的/component（最后一个斜线以及后面部分）去掉之后的字符串。这意味着在老的函数返回一个空字符串的情形下从 dirname()得到的是斜线或者一个点。

（7）pathinfo()函数。该函数返回文件路径的信息。声明如下：

```
array pathinfo ( string path)
```

pathinfo()返回一个数组包含有 path 的信息，包括以下的数组单元 dirname、basename 和 extension。示例如代码 5-21 所示。

【代码 5-21】 pathinfo.php

```
<?php
$path_parts = pathinfo("/www/htdocs/index.html");
echo $path_parts["dirname"] ."<br>";
```

```
echo $path_parts["basename"]."<br>";
echo $path_parts["extension"]."<br>";
?>
```

以上代码输出结果为：

```
/www/htdocs
index.html
html
```

（8）realpathf()函数。该函数返回规范化的绝对路径名。函数声明如下：

```
string realpath ( string path)
```

realpath()扩展所有的符号连接并且处理输入的 path 中的'/./', '/../'以及多余的'/'并返回规范化后的绝对路径名。返回的路径中没有符号连接，'/./'或'/../'成分。

realpath()失败时返回 FALSE，例如如果文件不存在的话。

（9）copy()函数。该函数复制文件。声明如下：

```
bool copy ( string source, string dest)
```

该函数将文件从 source 复制到 dest。如果成功则返回 TRUE，失败则返回 FALSE。

实例 5-19　文件复制

本实例演示如何进行文件复制，如代码 5-22 所示。

【代码 5-22】　copy.php

```
<?php
if (!copy("readfile.php", 'temp.bak'))
   print ("文件复制失败<br>\n");
else
    print("文件复制成功!");
?>
```

执行结果如下所示：

```
文件复制成功!
```

（10）disk_free_space()函数。该函数返回目录中的可用空间。声明如下：

```
float disk_free_space ( string directory)
```

该函数给出一个包含有一个目录的字符串，函数将根据相应的文件系统或磁盘分区返回可用的字节数。

实例 5-20　获取磁盘剩余空间

本实例使用 disk_free_space 函数获取指定磁盘或者目录的剩余空间，如代码 5-23 所示。

【代码 5-23】　disk_free_space.php

```
<?php
// $df 包含根目录下可用的字节数
$df = disk_free_space("/");
echo $df;
?>
```

执行结果如下：

```
12835323904
```

提示 本函数不能用于远程文件，被检查的文件必须通过服务器的文件系统访问。

（11）disk_total_space()函数。该函数返回目录的磁盘总大小。声明如下：

```
float disk_total_space ( string directory)
```

directory 参数给出一个包含有一个目录的字符串，函数将根据相应的文件系统或磁盘分区返回所有的字节数。

实例 5-21 获取磁盘总空间

本实例使用 disk_total_space 函数获取指定磁盘的总空间，如代码 5-24 所示。

【代码 5-24】 disk_total_space.php

```
<?php
// $df 包含 "/" 目录的磁盘大小
$df = disk_total_space("/");
echo $df;
?>
```

执行结果如下：

```
20003848192
```

（12）file_put_contents()函数。该函数将一个字符串写入文件。声明如下：

```
int file_put_contents ( string filename, string data [, int flags [, resource
context]])
```

和依次调用 fopen()、fwrite()以及 fclose()功能一样。本函数返回写入文件的字节数。flags 可以是 FILE_USE_INCLUDE_PATH 和/或 FILE_APPEND，但是使用 FILE_USE_INCLUDE_PATH 选项时要多加小心。

（13）fileatime()函数。该函数取得文件的上次访问时间。声明如下：

```
int fileatime ( string filename)
```

函数返回文件上次被访问的时间，如果出错则返回 FALSE。时间以 UNIX 时间戳的方式返回。

注意 一个文件的 atime 应该在不论何时读取了此文件中的数据块时被更改。当一个应用程序定期访问大量文件或目录时很影响性能。有些 UNIX 文件系统在加载时关闭 atime 的更新以提高这类程序的性能。

实例 5-22 获取文件的上次访问时间

本实例演示了如何通过 fileatime 函数获取文件的上次访问时间，如代码 5-25 所示。

【代码 5-25】 fileatime.php

```
<?php
// 输出类似：December 29 2002 22:16:23.
$filename = 'hello/student.txt';
if (file_exists($filename)) {
   echo "$filename 文件最近一次访问时间: " . date ("F d Y H:i:s.",
fileatime($filename));
}
```

```
?>
```

执行结果如下：

```
hello/student.txt 文件最近一次访问时间: December 04 2005 00:00:00.
```

（14）filemtime()函数。该函数取得文件修改时间，声明如下：

```
int filemtime ( string filename)
```

该函数返回文件上次被修改的时间，出错时返回 FALSE。

实例 5-23　获取文件的最近修改时间

本实例演示了如何获取文件的最近修改时间，如代码 5-26 所示。

【代码 5-26】　filemtime.php

```
<?php
// 输出: December 29 2002 22:16:23.
$filename = 'hello/student.txt';
if (file_exists($filename)) {
    echo "$filename 最近修改时间: " . date ("F d Y H:i:s.", filemtime($filename));
}
?>
```

执行结果如下：

```
hello/student.txt 最近修改时间: October 02 2005 13:56:44.
```

（15）stat()函数。该函数给出文件的信息，函数声明如下：

```
array stat ( string filename)
```

该函数获取由 filename 指定的文件的统计信息。如果 filename 是符号连接，则统计信息是关于被连接文件本身的，而不是符号连接。lstat()和 stat()相同，除了 stat 函数会返回符号连接的状态。如果出错，stat()返回 FALSE，并且发出一条警告。

返回一个数组包含的文件的统计信息，则该数组具有以下列出的单元，数组下标从 0 开始。除了数字索引之外自 PHP 4.0.6 以后还可以通过关联索引来访问。

stat()和 fstat()返回格式如表 5-2 所示。

表 5-2　　stat()和 fstat()返回格式

数字下标	关联键名（自 PHP 4.0.6）	说　明	数字下标	关联键名（自 PHP 4.0.6）	说　明
0	dev	设备名	7	size	文件大小的字节数
1	ino	Inode 号码	8	atime	上次访问时间（UNIX 时间戳）
2	mode	inode 保护模式	9	mtime	上次修改时间（UNIX 时间戳）
3	nlink	被连接数目	10	ctime	上次改变时间（UNIX 时间戳）
4	uid	所有者的用户 id	11	blksize	文件系统 IO 的块大小
5	gid	所有者的组 id	12	blocks	所占据块的数目
6	rdev	设备类型，如果是 inode 设备			

（16）fstat()函数。该函数通过已打开的文件指针取得文件信息，声明如下：

```
array fstat ( resource handle)
```

该函数获取由文件指针 handle 所打开文件的统计信息。此函数和 stat()函数相似，但 fstat 函数用于已打开的文件指针而不是文件名。

函数的示例如代码 5-27 所示。

【代码 5-27】

```
<?php

// 打开文件
$fp = fopen("/etc/passwd", "r");

// 取得统计信息
$fstat = fstat($fp);

// 关闭文件
fclose($fp);

// 只显示关联数组部分
print_r(array_slice($fstat, 13));

?>
```

输出结果如下：

```
Array
(
    [dev] => 771
    [ino] => 488704
    [mode] => 33188
    [nlink] => 1
    [uid] => 0
    [gid] => 0
    [rdev] => 0
    [size] => 1114
    [atime] => 1061067181
    [mtime] => 1056136526
    [ctime] => 1056136526
    [blksize] => 4096
    [blocks] => 8
)
```

（17）filetype()函数。该函数取得文件类型，声明如下：

```
string filetype ( string filename)
```

返回文件的类型。可能的值有 fifo，char，dir，block，link，file 和 unknown。如果出错则返回 FALSE。如果 stat 调用失败或者文件类型未知，filetype()还会产生一个 E_NOTICE 消息。

（18）is_dir()函数。该函数判断给定文件名是否是一个目录，声明如下：

```
bool is_dir ( string filename)
```

如果文件名存在并且为目录则返回 TRUE。如果 filename 是一个相对路径，则按照当前工作目录检查其相对路径。

（19）flock()函数。该函数进行文件锁定，声明如下：

```
bool flock ( int handle, int operation [, int &wouldblock])
```

PHP 支持以咨询方式（也就是所有访问程序必须使用同一方式锁定，否则它不会工作）锁定全部文件的一种轻便方法。

flock()操作的 handle 必须是一个已经打开的文件指针。operation 可以是以下值之一。

要取得共享锁定（读取程序），将 operation 设为 LOCK_SH。

要取得独占锁定（写入程序），将 operation 设为 LOCK_EX。

要释放锁定（无论共享或独占），将 operation 设为 LOCK_UN。

如果不希望 flock()在锁定时堵塞，则可以给 operation 加上 LOCK_NB。

flock()允许执行一个简单的可以在任何平台中使用的读取/写入模型，（包括大部分的 UNIX 派生版甚至是 Windows）。如果锁定堵塞（EWOULDBLOCK 错误码情况下），可将可选的第三个参数设置为 TRUE。

函数 flock 的示例如代码 5-28 所示。

【代码 5-28】　flock.php

```
<?php
$fp = fopen("hello/student.txt", "w+");
if (flock($fp, LOCK_EX)) { // 进行排它型锁定
   fwrite($fp, "Write something here\n");
   flock($fp, LOCK_UN); // 释放锁定
} else {
   echo "Couldn't lock the file !";
}
fclose($fp);
?>
```



由于 flock()需要一个文件指针，因此可能需要一个锁定文件来保护到打算通过写模式打开来截断的文件的访问（在 fopen()函数中加入"w"或"w+"）。

（20）is_uploaded_file 函数。该函数判断文件是否是通过 HTTP POST 上传的。

```
bool is_uploaded_file ( string filename)
```

如果 filename 所给出的文件是通过 HTTP POST 上传的，则返回 TRUE。这可以用来确保恶意的用户无法欺骗脚本去访问本不能访问的文件。

（21）move_uploaded_file()函数。该函数将上传的文件移动到新位置。声明如下：

```
bool move_uploaded_file ( string filename, string destination)
```

本函数检查并确保由 filename 指定的文件是合法的上传文件（即通过 PHP 的 HTTP POST 上传机制所上传的）。如果文件合法，则将其移动为由 destination 指定的文件。

如果 filename 不是合法的上传文件，不会出现任何操作，move_uploaded_file()将返回 FALSE。

如果 filename 是合法的上传文件，但出于某些原因无法移动，不会出现任何操作，move_uploaded_file()将返回 FALSE。此外还会发出一条警告。

如果目标文件已经存在，将会被覆盖。



当安全模式（safe mode）被激活，PHP 将检查将要操作的文件或者目录与正在执行的脚本是否有相同的 UID。

第 6 章　PHP 与面向对象程序设计

许多 PHP 的爱好者在学习过程中感到对 PHP 中类的概念较难理解和掌握，虽然知道类既然存在就有其存在的道理，但是由于平时接触和使用的机会较少，也就一略而过。其实，只要能够理解变量和函数这些 PHP 的基本概念，掌握类就不成问题。

6.1　面向对象编程知识点回顾

类是面向对象编程（OOP，Object Oriented Programming）的产物。那么，什么是面向对象编程呢？

面向对象技术的基本观点如下。

（1）客观世界由对象组成，任何客观实体都是对象，复杂对象可以由简单对象组成。

（2）具有相同数据和操作的对象可归纳成类，对象是类的一个实例。

（3）类可以派生出子类，子类除了继承父类的全部特性外还可以有自己的特性。

（4）对象之间的联系通过消息传递来维系。由于类的封装性，它具有某些对外界不可见的数据，这些数据只能通过消息请求，调用可见方法来访问。

下面分别介绍面向对象的几个核心元素：对象、封装、继承和多态。

6.1.1　对象

在面向对象技术中，对象是必不可少的基本成分。客观世界里的任何实体都可以被称之为对象，对象可以是具体的、有形的物，如人、汽车等；也可以是无形的事物或概念，如抽象的规则、计划或事件。复杂的对象可由相对比较简单的对象以某种方法组成，甚至整个世界也可以从一些最原始的对象开始，层层组合而成。

对象是一个封装数据属性和操作行为的实体。属性数据描述了对象的静态特征，即对象的状态；操作又称为方法或服务，它描述了对象执行的功能，操作可访问并修改私有的属性数据以改变对象的状态。一个对象可以有多个属性和多个服务，对象的属性和服务被结合成一个整体，对象的属性值只能由这个对象的服务存取。

系统中的对象在软件生命周期的各个阶段可能有不同的表示形式。例如，在分析与设计阶段对象表现为某种 OOA/OOD 方法给出的粗略定义，而在编程阶段则表现为用一种程序设计语言写出的详细而明确的源程序代码。

可以拿收音机为例建立一个对象模型，图 6-1 所示描述了一个收音机对象的例子。

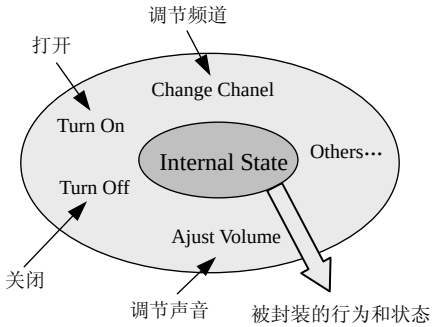


图 6-1　一个对象的实例图解

6.1.2　类

类是一组具有相同数据结构和相同操作的对象的集合。类是对象的抽象，它将一组数据属性和在数据上的一组合法操作抽象封装。类好比是一个对象模板，基于类就可以生成多个具有不同属性值的对象，在类的定义中，可以为某个属性指定一个值，作为它的默认值。

通常，把基于某个类生成的对象称为这个类的实例。客观世界实际存在的都是对象，而不是类。对象可使用类中提供的函数，对象的状态则包含在它的实例变量，即实例的属性中。

关于类和对象的关系，举例说明，假设有一个类“三角形”，为了简单起见，它的属性是 3 个顶点的坐标，操作有两个：“移动”和“判断一点是否在三角形内”。由类定义可以看出，类是对现实世界具有相同属性和操作的事物的抽象表示。从三角形类可以生成无数三角形对象，例如一个顶点坐标为（10，0）、（0，15）、（20，15）的三角形和一个顶点坐标为（50，50）（20，100）（100，150）的三角形。这两个三角形分别可以执行“移动”、“判断一点是否在三角形内”等类的操作。

6.1.3　继承

继承性是面向对象程序设计语言不同于其他语言的最主要的特点。继承是指子类可以自动拥有父类的全部属性与操作的机制。这种使用已存在的定义作为基础建立新定义的技术使得父类和子类之间能够共享数据和方法。

在对现实世界建模时，可以根据事物的共性抽象出一批基类，在基类的基础上事物可以根据个性添加自己的属性与操作，抽象出新的对象类。这些新类既有基类的全部属性与操作，又有自己的属性与操作，新类称之为基类的子类或派生类，基类称之为父类或超类。这种产生子类的方法就叫继承。

继承的形象描述如图 6-2 所示。

图 6-2 表示了父类 A 和它的子类 B 之间的继承关系，箭头从子类 B 指向父类 A 表示类 B 继承了类 A。子类 B 由继承部分 C 和增加部分 D 组成，C 部分是类 B 从类 A 继承的属性与操作，D 部分是类 B 自己特有的属性与操作。

继承性又分为单重继承和多重继承两类。单重继承时一个子类只能有一个父类；多重继承时一个子类可以有多于一个的父类。图 6-3 所示描述了单重继承（左图）和多重继承（右图）中父类和子类之间的关系。

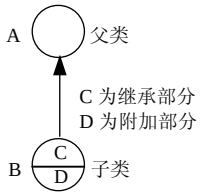


图 6-2　继承的描述

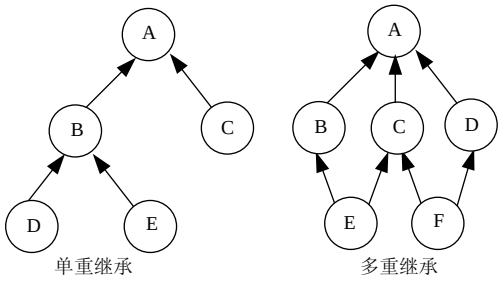


图 6-3　单重继承和多重继承的描述

由图 6-3 可以看出，单重继承构成的类之间的关系是一棵树，多重继承构成的类之间的关系是一个图。继承具有传递性。如图 6-3 中的左图，E 继承 C，而 C 继承 A，由继承的传递性，E 也继承了 A。所以 E 也是 A 的子类，称为 A 间接的子类，它不仅继承了类 C 的全

部属性与操作，也继承了类 A 的全部属性与操作。推广而言，一个子类除了具有自己的全部特性以外还具有该类上层全部基类的一切特性。

6.1.4 多态性

多态性是面向对象技术的一个重要概念。同一操作作用于不同的对象，可以有不同的解释，产生不同的执行结果，这就是多态性。多态性通常通过派生类重载基类中的同名函数来实现。

多态性分为如下两种。

（1）编译时的多态性。编译时的多态性是通过重载来实现的。系统在编译时，根据传递的参数、返回的类型等信息决定实现何种操作。

（2）运行时的多态性。运行时的多态性就是指直到系统运行时，才根据实际情况决定实现何种操作。

编译时的多态性提供了运行速度快的特点，而运行时的多态性则带来了高度灵活和抽象的特点。

多态性大致有以下 3 种实现方式。

（1）通过接口实现多态性。接口描述类需要实现的操作、属性和事件，以及每个操作需要接收和返回的参数类型，但将这些操作的特定实现留给实现接口的类去完成。多个类可实现相同的“接口”，而单个类可以实现一个或多个接口。这样实现某接口的类都有此接口定义的操作的不同实现方式。

（2）通过继承实现多态性。多个类可以从单个基类“继承”，子类不仅接收基类的所有方法、属性和事件，还可以重写基类的操作以提供不同的实现。继承实现的多态性使成员函数根据收到该调用的对象的类型产生不同的动作，类的使用者可以处理普遍性而让执行环境处理特殊性，提高了系统的可扩展性。

（3）通过抽象类实现的多态性。抽象类为多态的实现提供了极大的灵活性，因为它同时提供继承和接口的元素。抽象类本身不能实例化，必须被继承。抽象类的部分或全部成员可能未实现，该实现由继承类提供。已实现的成员仍可被重写，并且继承类仍可以实现附加接口或其他功能。

6.2 在 PHP 中使用类

上一节回顾了面向对象编程的一些基本概念。在接下来的章节中，将会介绍在 PHP 中如何进行简单的面向对象编程，首先就是学会如何创建类、属性和方法。

6.2.1 创建类

在 PHP 中，要创建一个类很简单，只需要关键字 class 即可，一个最简单的类如下所示：

```
class classname
{
}
```

在上面的代码中，classname 表示类名，类名的选择尽量有一定的意义，这样方便记忆和理解。

当然以上只是创建了一个空类而已，如果想它实现一定的功能，必须为该类添加属性和方法。

下面的代码为类添加了两个属性：

```
class classname
{
    var $myName;
    var $myAge;
}
```

创建类的属性也很简单，只需要在类中声明变量即可。声明了属性之后，可以获取该属性的值。

除了可以创建属性之外，还可以为类创建方法。为类创建方法也很简单，只需要创建函数即可，函数既可以有参数，也可以没有参数。如下面代码所示：

```
class classname
{
    var $myName;
    var $myAge;
    function getName()
    {
    }
    function getAge($arg1,$arg2)
    {
    }
}
```

在上面代码中，创建了两个类的方法，方法名称分别为 getName 和 getAge。可以往函数中添加代码以实现某种设定的功能。

代码 6-1 定义了一个名为 Cart 的类。

【代码 6-1】　class_1.php

```
<?php
class Cart
{
   var $items;  // 购物车中的项目
   // 把 $num 个 $artnr 放入车中
   function add_item ($artnr, $num)
   {
      //代码
   }
   // 把 $num 个 $artnr 从车中取出
   function remove_item ($artnr, $num)
   {
       //代码
   }
}
?>
```

在代码 6-1 中，定义了一个属性和两个方法。



提示　不能将一个类分开定义到多个文件或多个 PHP 标记对中，下面的代码是错误的。

```
<?php
class test {
?>
```

```
<?php
   function test() {
      print 'OK';
   }
}
?>
```



提示 名称 stdClass 已经被 Zend 使用并保留。不能在 PHP 代码中定义名为 stdClass 的类。函数名_sleep 和_wakeup 在 PHP 类中是魔术函数。除非想要与之联系的魔术功能，否则在任何类中都不能以此命名函数。PHP 将所有以_开头的函数名保留为魔术函数。除非想要使用一些见于文档中的魔术功能，否则建议不要在 PHP 中将函数名以_开头。

6.2.2 构造函数

有过类编程经验的读者一定知道，大多数的类都有构造函数。当一个类实例被调用时，将调用构造函数，通常在构造函数中将执行一些初始化的操作。

在类中声明构造函数，同声明其他函数一样，只是构造函数的名称是固定的，即函数名称必须为__construct()。这是在 PHP 5.0 中新增加的特性，在以前的版本中，构造函数的名称必须与类名相同。但是为了向下兼容，PHP 首先在类中搜索是否有__construct()函数，如果没有，则继续搜索是否含有与类同名的函数。

构造函数虽然可以由编程者主动调用，但是大部分时候，都是在创建类对象实例的时候自动调用。

声明构造函数的方法如下：

```
class classname
{
   function __construct($param)
   {
      //代码
   }
}
?>
```

构造函数可以传递参数，这些参数可以在调用类的时候传递。

6.2.3 析构函数

在 PHP4.0 以前，并没有提供析构函数，可以使用 register_shutdown_function()函数来模拟多数析构函数的效果。析构函数是一种当对象被销毁时，无论使用了 unset()或者简单的脱离范围，都会被自动调用的函数。析构函数允许在销毁一个类之前操作或者完成一些功能。

在 PHP 中，与构造函数的名称类似，一个类的析构函数名称必须是_destruct()。



提示 析构函数不能带有任何参数。

6.2.4 类的实例化

在声明了一个类之后，如果要使用该类，必须创建该类的一个实例。实际上也可以将该

类作为一个变量，在使用之前必须手动定义。定义该类实例，被称为创建一个实例或者实例化一个类。在 PHP 中，可以使用关键字“new”来创建一个类实例。

实例 6-1　创建一个简单的类

本实例创建了一个简单的类，然后创建了 3 个该类的实例，如代码 6-2 所示。

【代码 6-2】　firstclass.php

```
<?php
class myName
{
    function __construct($myName)  //构造函数
    {
        echo("我的名字是：$myName<br>");
    }
}

//下面创建类实例
$name1=new myName("小狗");
$name2=new myName("小猫");
$name3=new myName("小马");
?>
```

执行结果如图 6-4 所示。



图 6-4　类的构造函数

在以上代码中，“new myName（"小狗"）”中的参数“小狗”会被自动传递到构造函数，然后构造函数直接使用该参数。

提示　construct 前面是两个连续下划线。

6.2.5　使用类的属性

在一个类中，访问类成员变量有一个特殊的指针——$this。通过该指针，可以方便地访问类属性或者给类属性赋值。

如果类的属性名称为$myAttrib，则如果要设置该变量，只需要通过“$this->myAttrib”语句即可，如下面代码所示。

```
class classname
{
    var $myAttrib;   //定义属性
    function operate($myparam)
    {
        $this->myAttrib=$myparam;  //设置属性值为$myparam
        echo $this->myAttrib;  //输出属性值
    }
}
```

如果要在类的外部访问一个类的属性，需要确定该属性是否能被外部访问。因为在类中定义的属性，可以设置外界访问的权限。在后面章节将会介绍。如果像上面代码一样没有对属性$myAttrib 进行限制，则外界可以直接访问。访问方式如下：

```
$newclass=new classname();  //创建一个类 classname 实例
$newclass-> myAttrib="小马";   //给属性赋值
echo $newclass-> myAttrib;  //输出属性
```

一般来说，从类的外部直接访问类的属性并不是好的方法。面向对象方法的一个优点就是使用类的封装。可以通过使用_get()和_set()函数来实现对类属性的访问，如下面代码所示：

```
class classname
{
    var $myAttrib;   //定义属性
    function __get($attribName)  //获取属性值的函数
    {
        return $this->$ attribName;  //返回该属性值
    }
    function __set($attribName,$value)  //设置属性值的函数
    {
        $this->$attribName=$value;  //设置属性值
    }
}
```

以上代码完成了类中函数的属性设置和获取。



注意 __get 函数只有一个参数，该参数用来传递属性的名称。__set 函数有两个参数，第一个参数为属性名称，第二个参数为要设置的属性的值。可以通过这两个函数来检查和设置任何属性的值。

当然在访问和设置属性的时候不需要直接使用这两个函数，下面简单介绍其工作原理。

首先需要创建一个类实例，如下所示：

```
$newclass=new classname();  //创建一个类 classname 实例
```

如果要设置属性“myAttrib”的值，可以使用如下代码：

```
$newclass-> myAttrib="小马";
```

设置了__get()和__set()函数之后，代码将首先间接地调用__set()函数，将__set()函数的$attribName 参数设置成“myAttrib”，而参数$value 则被设置成“小马”。可以在__set()函数中编写代码进行任何需要的错误检查。

同样，如果要获取一个属性的值，可以使用下面的代码：

```
$myVal=$newclass-> myAttrib;
```

以上语句首先调用__get()函数，该函数的$attribName 值被设置成“myAttrib”。

细心的读者会发现，以上访问属性的代码同没有__get()和__set()函数的访问代码一样，但是实质上确保了整个类的属性只有一个访问入口。

6.2.6 类的访问控制

在 PHP5 中，引入了类的访问控制符，这样可以控制类的属性和方法的可见性。在 PHP5 中支持以下 3 种访问控制符。

（1）public 控制符。该控制符是默认的，如果不指定一个属性的访问控制，则默认是 public。public 表示该属性和方法在类的内部或者外部都可以被直接访问。

（2）private 控制符。该控制符说明属性或者方法只能够在类的内部进行访问。如果没有使用__get()和__set()方法，则可能会对所有的属性都使用这个关键字，也可以选择使得部分方法为私有的。注意，私有属性和方法不能被继承。

（3）protected 控制符。该控制表示被标记的属性或者方法只能在类的内部进行访问。它也存在于任何子类中。

下面的代码添加了访问控制符：

```
class classname
{
    public  $myAttrib;   //定义属性
    public function __get($attribName)  //获取属性值的函数
    {
        return $this->$ attribName;  //返回该属性值
    }
    public function __set($attribName,$value)  //设置属性值的函数
    {
        $this->$attribName=$value;  //设置属性值
    }
}
```

上述代码中，每个类成员都具有一个访问修饰符，说明它们是公有的还是私有的。可以不添加 public 关键字，因为它是默认的控制符。

6.2.7　类方法的调用

前面介绍了类属性的访问和设置。类方法的访问和调用同类属性基本类似，假如一个类声明如下：

```
class classname
{
    var $myAttrib;   //定义属性
    function func_1($param)  //函数 func_1
    {
        //函数代码
    }
    function func_2($param1, $param2)  //函数 func_2
    {
        //函数代码
    }
    function func_3($param1, $param2)  //函数 func_2
    {
        //函数代码
        return backvalue;
    }

}
```

要调用以上类的方法，首先要创建一个类实例 ，如下所示：

```
$newclass=new classname();  //创建一个类 classname 实例
```

调用以上类的 3 个函数，非常简单，如同调用属性一样，如下所示：

```
$newclass->func_1("123");  //调用函数 func_1
$$newclass->func_2("123","abc");  //调用函数 func_2
```

```
$myvalue=$newclass->func_3("123","abc");  //调用 func_3 并获取返回值
```

6.2.8 类的应用实例

下面简单地创建一个完整的类。该类表示一个用户的购物栏，当用户添加商品之后，购物栏中便会增加对应商品和数量。

实例 6-2 创建一个完整的类

本实例创建一个完整的类，该类中有构造函数，有属性、方法等，如代码 6-3 所示。

【代码 6-3】 completeclass.php

```
<?php
class Cart {
   var $items;  // 购物车中的物品
   var $name;

   // 将 $num 个 $artnr 物品加入购物车
    function __construct($myName)  //构造函数
    {
        echo("$myName 的购物车，购物过程如下：<br>");
        $this->name=$myName;
    }

    //添加商品
   function add_item($artnr, $num) {
      $this->items[$artnr] += $num;
       echo "<br>添加 $num 个 $artnr,$artnr 的总数为 " .$this->items[$artnr];
   }

   // 将 $num 个 $artnr 物品从购物车中取出
   function remove_item($artnr, $num) {
      if ($this->items[$artnr] > $num) {
         $this->items[$artnr] -= $num;
           echo "<br>放回 $num 个 $artnr,$artnr 的总数为 ".
$this->items[$artnr];
         return true;
      } elseif ($this->items[$artnr] == $num) {
         unset($this->items[$artnr]);
           echo "<br>放回 $num 个 $artnr,$artnr 的总数为 0";
         return true;
      } else {
         return false;
      }
   }
}

$myCart=new Cart("张三");
$myCart->add_item("苹果",9);
$myCart->add_item("香蕉",8);
$myCart->remove_item("苹果",2);
```

```
$myCart->remove_item("香蕉",3);
?>
```

执行结果如图 6-5 所示。

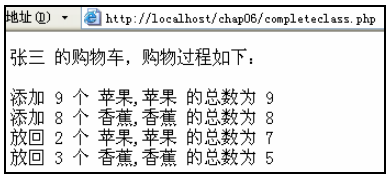


图 6-5　代码 completeclass.php 的执行结果

6.3　面向对象编程的高级技术

前面介绍了类的基本使用技巧，接下来介绍面向对象编程的一些高级技术，如类的继承、类的多态等。

6.3.1　继承的实现

6.1.3 小节已经介绍了继承的基本概念，下面将通过实例来学习继承的应用。继承在 PHP 中很容易实现，只要使用“extends”关键字。

首先以前面创建的 completeclass.php 文件中的 Cart 类为基类，该类有一个属性和两个方法。接下来通过关键字“extends”来创建一个子类。

实例 6-3　类的继承

本实例演示如何从一个基类继承一个子类，并且使用基类的方法和属性，如代码 6-4 所示。

【代码 6-4】　extends.php

```
<?php
class location_Cart extends Cart {
   var $location;  //子类中声明属性

   function set_location ($myloacation) {  //子类中声明方法
      $this->location = $myloacation;
   }
}
?>
```

上述代码定义了名为 location_Cart 的类，该类拥有 Cart 类的所有变量和函数，加上附加的变量$loacation 和一个附加函数 set_location ()。现在，以正常的方式创建了一个有名字的购物车，并且可以设置并取得该购物车的主人的地址。而基类的函数依旧可以在子类中使用。

```
<?php
//创建子类
$myCart=new location_Cart("张三");  //构造函数设置主人
//调用子类方法
$myCart->set_location("上海");  //设置地址

echo $myCart->name. "的地址是" . $myCart->location;   //显示地址
$myCart->add_item("苹果",9);  //调用基类方法
```

```
$myCart->add_item("香蕉",8);
$myCart->remove_item("苹果",2);
$myCart->remove_item("香蕉",3);
?>
```

执行结果如图 6-6 所示。

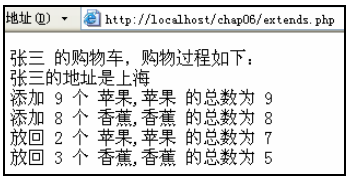


图 6-6 类的继承的实现

如果有必要，还可以创建以子类 location _Cart 为父类的子类。

注意 继承是单方向的，子类可以从父类或者超类继承特性，而父类不能从子类继承特性。

PHP 现在还不支持多重继承，所以不能从两个或两个以上类派生出新的类来。

提示 PHP4 和 PHP5 中，一个对象都仅能继承一次，多重继承是不被支持的。不过，在 PHP5 中产生了一个新的名词：接口。接口是一个没有具体处理代码的特殊对象，它仅仅定义了一些方法的名称及参数，此后，对象就可以方便地使用'implement'关键字把需要的接口整合起来，然后再加入具体的执行代码。

如下面定义的接口代码：

```
<?php
interface displayable {
    function display();
}
interface printable {
    function doprint();
}

class foo implements displayable,printable {
    function display() {
        // 函数代码
    } function doprint() {
        // 函数代码
    }
}
?>
```

以上接口的定义对提高代码的可读性有很大的帮助，对象 foo 包含了 displayable 和 printable 两个接口，这时就可以清楚地知道，对象 foo 一定会有一个 display()方法和一个 print()方法，只需要去了解接口部分，就可以轻易地操作该对象而不必去关心对象的内部是如何运作的。

6.3.2 多态的实现

多态是对象的一种能力，它可以在运行时刻根据传递的对象参数，动态地决定调用哪一

个对象的方法。例如，如果有一个 figure 的类，它定义了一个 draw 的方法，并且派生了 circle 和 rectangle 类，并在派生类中覆盖了 draw 方法。同时可能还有一个函数 draw，希望使用该函数传递不同个数的参数。如果有多态性，则调用哪个 draw 方法就依赖于传递给这个函数的对象类型。

实例 6-4　类的多态

本实例演示了类的多态，如代码 6-5 所示。

【代码 6-5】　multi.php

```
<?php
//首先创建了一个基类
class figure
{
    var $a;
    function draw($x,$y)
    {
        echo "横坐标是: $x,纵坐标是 $y <br>";
        //其他代码
    }
}
/*以上创建了一个类，然后接下来创建一个子类 circle，
该类在继承父类的同时，同时对父类的函数 draw 进行了重载*/
class circle extends figure
{
    function draw($x,$y)
    {
        echo "圆心是: $x,半径是: $y <br>";
    }
}


//另外还派生了另外一个类 rectangle 类，如下所示。
class rectangle extends figure
{
    function draw($x,$y,$z)
    {
        echo "长方体的长: $x,宽: $y,高: $z <br>";
    }
}

//最后还有一个派生类，该类没用重载父类的 draw 函数。
class line extends figure
{
    function drawline($x,$y)
    {
        echo "画直线,起点是 $x,终点是 $y <br>";
    }
}
```

```
//创建 figure 类实例
$a=new figure();
$a->draw(10,100);  //调用父类的 draw 函数

//创建 circle 类实例
$b=new circle();
$b->draw(100,100);   //调用 circle 类重载后的 draw 函数

//创建 rectangle 类实例
$c=new  rectangle ();
$c->draw(100,100,1000);   //调用 rectangle 类重载后的 draw 函数

//创建 line 类实例
$d=new  line ();
$d->draw(10,100);   //由于 line 类没有重载，所以调用了父类的 draw 函数
?>
```

执行结果如图 6-7 所示。



图 6-7　类的多态

6.3.3　作用域分辨运算符

“::”（作用域分辨）运算符可以在没有声明任何实例的情况下访问类中的函数或者基类中的函数和变量。

实例 6-5　“::”运算符

本实例演示了“::”运算符的使用，如代码 6-6 所示。

【代码 6-6】　class_1.php

```
<?php
class A {
    function example() {
        echo "我是基类的函数 A::example().<br />";
    }
}

class B extends A {
    function example() {
        echo "我是子类中的函数 B::example().<br />\n";
        A::example(); //调用父类的函数
    }
}

// A 类没有对象实例，直接调用其方法 example
A::example();
```

```
// 建立一个 B 类的对象
$b = new B;

//调用 B 的函数 example
$b->example();
?>
```

执行结果如图 6-8 所示。

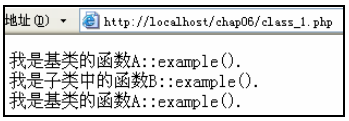


图 6-8　类的::运算符

上面的例子调用了 A 类的函数 example()，但是这里并不存在 A 类的对象，因此不能用$a->example()或者类似的方法调用 example()。反而将 example()作为一个类函数来调用，也就是说，作为一个类自身的函数来调用，而不是这个类的任何对象。

这里有类函数，但没有类的变量。实际上，在调用函数时完全没有任何对象。因而一个类的函数可以不使用任何对象（但可以使用局部或者全局变量），并且可以根本不使用$this 变量。

上面的例子中，类 B 重新定义了函数 example()。A 类中原始定义的函数 example()将被屏蔽并且不再生效，除非使用“::”运算符来访问 A 类中的 example()函数，如 A::example()。就此而论，对于当前对象，它可能有对象变量。因此可以在对象函数的内部使用$this 和对象变量。

6.3.4　parent 关键字

编程时，经常会在子类的代码中访问基类的变量和函数，尤其是派生类非常精炼或者基类非常专业化的时候。此时不要用代码中基类真正的名字，应该用特殊的名字“parent”，它指的是派生类在 extends 声明中所指的基类（父类）的名字。这样做可以避免在多个地方使用基类的名字。如果继承树在实现的过程中要修改，只要简单地修改类中 extends 声明的部分即可。这样可以提高代码的可读性以及可维护性。

实例 6-6　使用 parent 关键字

本实例演示了如何通过 parent 关键字访问基类的方法和属性，代码 6-7 所示。

【代码 6-7】　parent.php

```
<?php
class A {
   function example() {
      echo "I am A::example()<br />";
   }
}

class B extends A {
   function example() {
      echo "I am B::example()<br />";
```

```
        parent::example();
    }
}

$b = new B;
// 这将调用 B::example()，而它会去调用 A::example()。
$b->example();
?>
```

执行结果如图 6-9 所示。

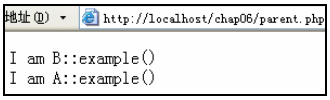


图 6-9　parent 关键字

6.3.5　final 关键字

在 PHP5 中引入了 final 关键字。当在一个函数声明前使用该关键字时，这个函数将不能被任何子类重载，如下面的一段代码：

```
<?php
class BaseClass {
   public function test() {
      echo "BaseClass::test() called\n";
   }

   final public function moreTesting() {
      echo "BaseClass::moreTesting() called\n";
   }
}

class ChildClass extends BaseClass {
   public function moreTesting() {
      echo "ChildClass::moreTesting() called\n";
   }
}
// 产生错误 Fatal error: Cannot override final method BaseClass::moreTesting()
?>
```

同样，如果不想让一个类被继承，也可以使用 final 关键字来声明一个类，如下代码所示：

```
<?php
final class BaseClass {
   public function test() {
      echo "BaseClass::test() called\n";
   }

   // Here it doesn't matter if you specify the function as final or not
   final public function moreTesting() {
      echo "BaseClass::moreTesting() called\n";
   }
}
```

```
    class ChildClass extends BaseClass {
    }
    //产生错误 Fatal error: Class ChildClass may not inherit from final class
(BaseClass)
    ?>
```



提示　如果一个类的属性或者方法被 private 关键字修饰，则该属性或者方法不能被继承。如果一个属性或者方法被 protected 关键字修饰，则它在类的外部不可见，但是可以被继承。

6.3.6　static 关键字

static 关键字适用于允许在未经初始化类的情况下调用类的属性和方法。该方法类似于前面介绍的“::”运算符。

实例 6-7　使用 static 关键字

本实例介绍 static 关键字的应用，如代码 6-8 所示。

【代码 6-8】　static.php

```
    <?php
    class Foo  //定义基类
    {
        //使用 static 关键字定义变量
       public static $my_static = 'foo';

       public function staticValue() {
          return self::$my_static;
       }
    }

    class Bar extends Foo  //创建基类
    {
       public function fooStatic() {
          return parent::$my_static;  //返回基类的变量$my_static
       }
    }

    //打印基类的$my_static 变量
    //虽然没有创建类实例，但是可以直接访问 static 变量
    print 'Foo::$my_static 结果为'.Foo::$my_static . "<br>";

    $foo = new Foo();  //创建基类实例
    print $foo->staticValue() . "<br>";  //通过方法放回 static 变量
    print $foo->my_static . "<br>";     // $my_static 为未定义的属性，所以为空

    //通过子类访问$my_static
    print Bar::$my_static . "<br>";
    $bar = new Bar();
```

```
print $bar->fooStatic() ;  //返回父类的 static 变量
?>
```

执行结果如图 6-10 所示。



图 6-10　static 关键字的应用

第 7 章　错误调试与异常处理

有过编程经验的人肯定知道，无论在编程的时候多细心或者多认真，都会在程序中留下或多或少的各种错误。因此当一个系统完成之后，对系统进行调试是软件开发中不可缺少的步骤。另外，为了尽量减少各种错误的出现，可以在代码中进行各种异常处理。

7.1　错 误 类 型

在介绍错误调试之前，首先介绍在编程过程中会经常遇到错误类型。

无论哪一种编程语言，常会遇到以下几种错误。

（1）语法错误。该错误是 PHP 在脚本做语法解析的时候所捕捉并警告的错误。

（2）运行时错误。该错误相比语法错误，不容易被发现，通常是在没有语法错误的情况下发生的，它可能出现在脚本交互的过程中或者其他的事件或者条件下。

（3）逻辑错误。这种错误通常是最难发现和清除的，通常出现在程序的概念或者逻辑中，因此叫做逻辑错误。对 PHP 而言，程序运行是正常的，而且也会产生执行结果，但是结果并不是程序员预期的。最常见的一个例子就是，在编程中写错了一个变量名称，而该变量名称并不是编程人员想跟踪的，这样产生的结果，当然不会是正确的。

下面通过一些实例来分析和处理这些常见的错误。

7.1.1　语法错误

语法错误表示 PHP 解释器因为脚本中的一些错误语法而不能正确解释该脚本。在所有的错误类型中，语法错误是最常见，同时也是危险性最小的，因为它们通常是因为编程人员疏忽引起的。

常见的引起语法错误的原因如下：

（1）缺少分号或者引号；

（2）关键字输入错误或者缺少逻辑结构；

（3）括号不匹配，如大括号、圆括号以及方括号；

（4）忘记使用变量前面的美元符号；

（5）错误地转义字符串中的特殊字符。

1．缺少分号

缺少分号是语法解析中出现概率最高的错误，如代码 7-1 所示。

【代码 7-1】　debug_1.php

```
<?php
    $a=1
```

```
    $b=6;
    echo "i love php";
?>
```

执行代码 debug_1.php，结果如图 7-1 所示。



图 7-1　缺少分号的错误提示

修改以上错误非常简单，只需要在第一句代码“$a=1”后面添加一个分号即可。

2．缺少引号

第 2 种常见的引起错误的原因是缺少引号，如单引号或者双引号等，如代码 7-2 所示。

【代码 7-2】　debug_2.php

```
<?php
    $a=1;
    $b=6;
    echo "i love php;
?>
```

代码 7-2 的执行结果如图 7-2 所示。



图 7-2　缺少引号的错误

修复以上错误很容易，只需要将“echo "i love php;”改成“echo "i love php";”即可。

另外一种缺少引号的情况是在输出字符串的时候，两端都忘记加引号，如代码 7-3 所示。

【代码 7-3】　debug_3.php

```
<?php
    $a=1;
    $b=6;
    echo i love php;
?>
```

执行结果如图 7-3 所示。

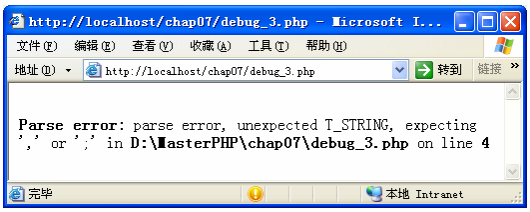


图 7-3　两端缺少引号的错误

3．缺少关键字或者逻辑结构

首先看代码 7-4：

【代码 7-4】　debug_4.php

```
<?php
    $a=1;
    do
    {
        echo "the number is $a";
        $a++;
    }
?>
```

代码 7-4 的输出结果如图 7-4 所示。



图 7-4　缺少关键字 while 的错误

从错误说明可以看出，解释器要求提供“T_WHILE”，也就是说可能在代码中缺少 While 关键字，因为 do…while 才是一个完整的循环语句。要想修改错误，只需要修改成如下代码：

```
<?php
    $a=1;
    do
    {
        echo "the number is $a";
        $a++;
    }while($a<10)
?>
```

4．缺少括号

在代码中可能需要用到很多括号，如大括号、圆括号以及中括号，当括号层数比较多的时候，有可能会引起括号不匹配。

代码 7-5 就是由括号不匹配导致的错误。

【代码 7-5】　debug_5.php

```
<?php
    $a=1;
```

```
    $b=2;
    $c=3;
    $d=4;
    if (($a>$b) and ($a>$c)) or ($c>$d)
    {
        echo "条件成立!";
    }
    else
    {
        echo "条件不成立!";
    }
?>
```

代码 7-5 的执行结果如图 7-5 所示。

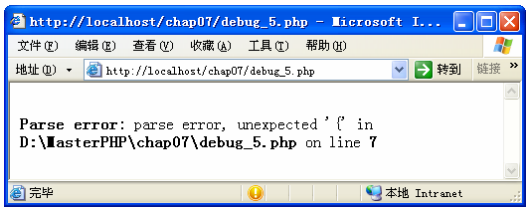


图 7-5 缺少括号

从图中的提示可以看出，大括号“{”后面有问题。实际上大括号没什么问题，因此可能是后面的括号出现问题。仔细检查会发现 if 语句的圆括号不平衡，只需要改成“if (($a>$b) and ($a>$c)) or ($c>$d))”，代码就可以正确执行了。

5．缺少美元符号

在 PHP 中，需要在变量前面添加美元符号，否则可能会引起解析错误，如代码 7-6 所示。

【代码 7-6】 debug_6.php

```
<?php
    for($i=0;i<100;$i++)
    {
        echo "i am $i";
    }
?>
```

代码 7-6 的执行结果如图 7-6 所示。

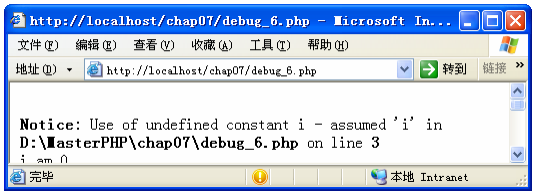


图 7-6 缺少美元符号

从图 7-6 中可以看出，由于“i<100”缺少美元符号，所以提示 i 没有定义，同时该程序进入无限循环状态直到服务器终止执行。要修正改程序，只需要将“i<100”改成“$i<100”即可。

以上介绍了一些常见的语法错误及修改方法。对于语法错误，只要编程人员根据编译器

的提示以及在书写代码的时候认真仔细一些，就很容易解决。

7.1.2　运行错误

运行错误相对于语法错误而言，更加复杂一些。通常运行错误是在语法没有错误的情况发生的。常见的运行错误有以下几种类型：

（1）被访问的数据库或者数据库表不存在；

（2）包含的文件或者文件路径不存在；

（3）PHP 服务器进程没有访问某个文件的权限；

（4）检查输入数据失败；

（5）连接到网络服务；

（6）调用不存在的函数。

下面通过实例来分析以上的各种运行错误。

1．包含的文件不存在

首先看代码 7-7 所示。

【代码 7-7】　debug_7.php

```
<?php
require ("debug_100.php");
?>
```

从表面上看，代码 7-7 没有任何错误，但是执行后会发现提示错误。执行结果如图 7-7 所示。

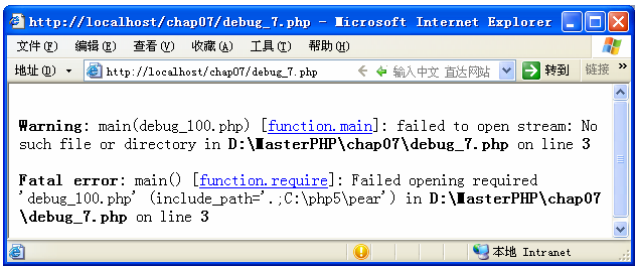


图 7-7　包含文件不存在

在前面章节已经学习了 include()和 require()函数，这两个函数可以包含其他的 PHP 文件。在 php.ini 文件中，有一个 include_path 选项，如果代码试图包含一个不在 include_path 路径中的文件或者该文件不存在，则会提示。

以上代码中由于 debug_100.php 文件不存在，所以提示错误，如果将其改成 debug_1.php，则不会产生错误。

2．计算式错误

首先查看代码 7-8。

【代码 7-8】　debug_8.php

```
<?php
    $a=1;
    $b=0;
```

```
    $c=$a/$b;
?>
```

代码 7-8 很容易理解，被除数是不能为 0 的，所以运行该代码会出现错误，如图 7-8 所示。



图 7-8　计算式不合法

虽然实际编程中不会这么写，但是很多时候“$b”可能是由用户手工输入的，如果不做处理，则可能导致运行错误。

3．文件操作与权限

编程人员在使用 fopen 函数进行文件操作的时候，如果没有考虑到被操作文件的权限，可能会导致运行错误。代码 7-9 试图向一个只读文件写数据。

【代码 7-9】　debug_9.php

```
<?php
    $fp=fopen("temp.txt","w");
    fwrite($fp,"hello world");
    fclose($fp);
?>
```

由于 temp.txt 文件被设置成只读文件，因此如果强行向此文件写入数据，则会提示错误，如图 7-9 所示。

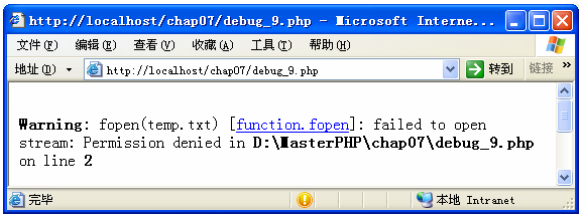


图 7-9　文件操作失败

解决这种问题的方法是首先获得该文件的属性判断，是否为可写，如果坚持要写，可以修改其权限。

另外由于 fopen 函数有返回值，因此根据其返回值来操作也是明智之举，如果返回值为 FALSE，则应该停止向文件写入数据。

4．调用不存在的函数

很多时候，由于编程人员疏忽，会调用不存在的函数。有些时候，内置函数也会出现命名不一致的情况。不但在当前脚本里容易调用自己编写的不存在的函数，而且其他地方也容易这样调用。如果传递给函数的参数个数不正确，也会发生错误。

【代码 7-10】　debug_10.php

```
<?php
    strstr();
```

```
?>
```

由于没有给 strstr 函数传递参数，所以会提出警告，如图 7-10 所示。

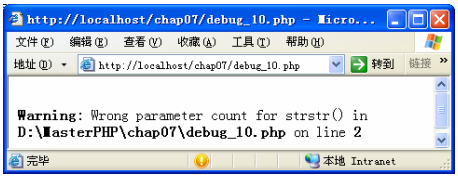


图 7-10　函数调用错误

5．连接数据库错误

数据库编程是 PHP 中最重要的内容，关于数据库的错误处理信息，请参考数据库编程章节。

6．连接到网络服务器

网络编程也是 PHP 中重要的内容。关于网络编程中的常见运行错误，将会在网络编程章节中介绍。

7.1.3　逻辑错误

逻辑错误是最难发现和清除的错误类型。这类错误的代码完全正确，而且也是按照正确的程序逻辑执行的，但是结果却不是编程人员预计的结果。

尽管捕捉某些逻辑错误很容易，因为它们属于可以查出的那些常见的错误类型，但是大多数逻辑错误并不真正属于某个特定的错误类型 。按照定义，产生错误的逻辑就是编程人员创建用来解决某个具体问题的逻辑，因此它对编程人员的代码常常是惟一的。一般来说，纠正逻辑错误很容易，但是发现逻辑错误是非常困难的。

假如在一个电子商务网有一个让用户填写数据的文本框，如果编程人员不加任何限制，则用户可能输入负数，如果这样可能会将 10 倍于该物品的价钱退还到客户信用卡中。

下面通过几个简单实例来说明逻辑错误。

1．计数错误

计数错误通常发生在数组编程中。如编程人员把值存储在一个数组的全部 6 个元素中，可是忽略了数组的索引是从 0 开始的，而将数据存进了元素 1 到元素 6 中，而索引为 0 的元素没有获得赋值。

另外，在循环计数的过程中，也容易产生错误，大部分的计数错误，偏差可能只有 1。

代码 7-11 所示是一个循环语句。

【代码 7-11】　debug_11.php

```
<?php
    $times=10;
    for($i=1;$i<$times;$i++)
    {
        echo "循环第 $i 次";
    }
?>
```

代码 7-11 的执行结果如图 7-11 所示。

在代码 7-11 中，编程人员原计划是想让语句循环输出 10 次，但是结果只输出了 9 次。原因很简单，变量$i 从 1 开始一直到 9 都符合限制条件“$i<times”，而到 10 的时候，则不

符合，停止循环，因此总共输出了 9 次。

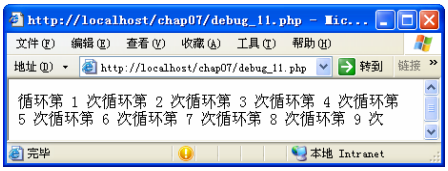


图 7-11　计数错误

修改这种错误很容易，只需要将“$i<times”改成“$i<=times”即可。

2．不测试返回值

在 PHP 编程中，调用有返回值的函数的时候，应尽量先判断其返回值，或者说要提高警惕。逻辑错误很多时候都是由于没有测试返回值造成的。

在 PHP 中，许多最常用的函数都是依赖于最初级的错误报告系统。如果函数工作正常，则会返回一些有用的数据，而如果失败一般会返回 FALSE。但是有时候正确的返回值和错误的返回值也有可能冲突。

如下面的代码：

```
substr("abcdef","a");
```

以上代码查询字符“a”，因为字符“a”在原始字符串“abcdef”中索引为 0，则以上语句返回 0。

在看下面的代码：

```
substr("abcdef","g");
```

以上代码查询字符“g”，因为原始字符串“abcdef”中没有“g”，因此该函数返回 FALSE。但是在 PHP 中，返回值 FALSE 也是默认为 0，因此在检测返回值的时候，为了严格区分，应该使用恒等运算符来判断。

总之，逻辑错误不是由任何一种代码失败导致的，而是程序员没有正确地编写代码来让计算机做人们期望的事情。结果，错误就不能自动被检测到。计算机也不能告诉编程人员出现了错误，更不会给一段代码告诉问题出在什么地方。逻辑错误只有通过适当的调试才能检测到。在调试的时候，首先要消除暗含在脚本中的假设，然后彻底测试每种可能有效或者无效的输入，确认这些输入都得到正确的结果。

7.1.4　设置错误报告级别

在 php.ini 中，可以设置服务器对错误的报警级别。在默认的情况下，PHP 将报告除了通知之外的所有错误。

错误报告级别是通过一些预定义的常量来设置的，如表 7-1 所示。

表 7-1　　错误报告常量

值	名　称	说　明
1	E_ERROR	报告运行时的致命错误
2	E_WARNING	报告运行时的非致命错误
3	E_PARSE	报告解析错误
8	E_NOTICE	报告通告、注意，表示所做的事情可能是错误的
16	E_CORE_ERROR	报告 PHP 引擎启动失败
32	E_CORE_WARNING	报告 PHP 引擎启动时非致命错误

续表

值	名　称	说　明
64	E_COMPILE_ERROR	报告编译错误
128	E_COMPILE_WARNING	报告编译时出现的非致命错误
256	E_USER_ERROR	报告用户触发的错误
512	E_USER_WARNING	报告用户触发的警告
1024	E_USER_NOTICE	报告用户触发的通告
2047	E_ALL	报告所有的错误和警告
2048	E_STRICT	报告不赞成的用法和不推荐的行为

每个常量都表示一种错误类型，错误可以被报告也可以被忽略。比如指定错误报告级别为 E_ERROR，则只有出现致命错误才会报告。这些常量可以用二进制算法结合起来，产生不同的错误报告级别。

在 php.ini 文件中，默认的报告是除了通知之外的所有错误，如下语句设定：

```
error_reporting  =  E_ALL & ～(E_NOTICE)
```

在上面的表达式中，符号“&”表示几个条件之间的并列，而“～”表示否定，即 NOT 的含义。

如果想忽略警告信息，可以设定如下语句：

```
error_reporting  =  E_ALL & ～(E_NOTICE) & ～(E_WARNING)
```

在 php.ini 文件中，和错误相关的设置有如下几个：

（1）error_reporting，设定错误报告级别；

（2）display_errors，是否显示错误报告，设置为 ON 则打开，设置为 OFF 则关闭所有错误提示；

（3）log_errors，默认设置为 OFF，是否记录错误日志；

（4）track_errors，默认设置为 OFF，该选项可以帮助解决代码中的错误，而不是让 PHP 提供其默认的功能。

7.2　错 误 调 试

前面大致介绍了错误的基本类型以及表现形式，下面简要介绍常用的调试技术。

7.2.1　调试过程

既然程序可能存在各种各样的错误，调试就显得尤为必要。虽然调试不是一门精确的科学，但是有效的调试方法可以大大缩短查找错误的时间。

在发现有错误的时候，首先要估计这些错误最可能是由哪些环节引起的，然后针对性地采用一些办法来发现错误原因并修正。

常用的调试过程如下：

（1）再现错误；

（2）估计可能的出错原因；

（3）查找错误源；

（4）改正错误（注意改动前要备份）；

（5）再次测试被修改的代码。

7.2.2 常用的调试手段

了解了调试过程之后，接下来介绍一些常用的调试手段。

（1）增加中间变量或者跟踪变量。当程序结果与预期的结果不一致的时候，可以通过增加中间变量，或者输出一些相关的变量值，发现错误根源。

（2）注释掉部分代码。这是最常见的调试办法，当出现错误的时候，如果不知道错误可能出现在什么地方，此时可以采用分段注释。当注释掉一段代码后，如果还发现同样的错误，则代码可能出现在没有被注释的部分。接着恢复被注释的代码，继续注释另外一段代码，直到不出现该错误，这样可以确认错误可能出现在最后被注释的代码段中。

（3）通过调试器来单步调试，这样可以跟踪整个程序的执行过程，发现是否有此应该被执行的函数没有被执行，或者变量赋值错误等各种导致错误的原因。

第 8 章　MySQL 数据库

数据库是任何一种网络编程中最重要的部分。如果要通过 ASP、JSP 以及 PHP 等语言来实现一个功能比较完善的系统，数据库是不可缺少的。

本章将重点讲解 MySQL 的基本知识以及常用操作技巧。有关关系数据库的设计理论，读者可以参考相关的书籍。

虽然本书重点介绍 MySQL，但是实际上 PHP 能够支持的数据库种类非常多。

8.1　MySQL 应用基础

在第 1 章已经详细介绍了 MySQL 的获取、安装以及启动等。如果读者已经成功安装了 MySQL 数据库，则可以开始本章的学习了。

8.1.1　登录到数据库

用户在客户端可以连接本地机器或者远程机器的 MySQL 数据库服务器。连接服务器的格式如下：

```
mysql -h 主机名 -u 用户名 -p 密码
```



注意　　-h 和主机名，-u 和用户名以及-p 和密码之间可以有空格，也可以没空格。

比如连接本地机器的 MySQL 数据库，可以用如下语句：

```
mysql -uroot -p123456
```

下面来测试该语句。为了方便测试，这里采用从 Windows 命令行登录的方法。（注意，为了能够从命令行登录 MySQL，需要在安装的时候进行设置，具体参见第 1 章）。选择菜单【开始→运行】，在弹出的对话框输入“cmd”，然后回车，可以进入命令行模式，如图 8-1 所示，然后输入命令“mysql –uroot –p123456”，可以直接登录到 MySQL 数据库。

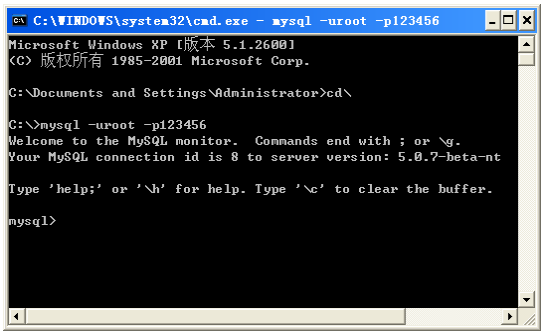


图 8-1　Windows 命令行模式登录 MySQL

图 8-1 所示表示已经登录成功。在输入语句的时候，也可以先不输入密码，则系统会提示输入密码，如输入以下命令：

```
mysql -uroot -p
```

执行过程如图 8-2 所示。

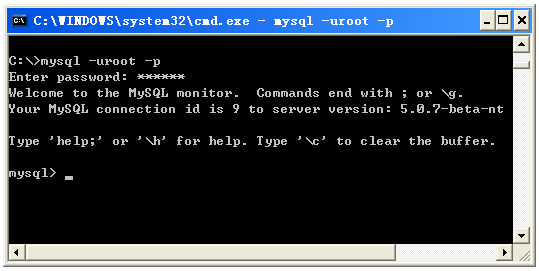


图 8-2 提示输入密码

由于在命令“mysql –uroot –p”中没有事先输入密码，则回车后会显示“Enter password:”提示输入密码。

如果要连接远程计算机，则需要加上 IP 地址或者机器名（域名）。如输入以下地址可以连接远程地址：

```
mysql -h127.0.0.1 -uroot -p123456
```

其中“127.0.0.1”是机器地址，这里为方便测试，使用了本地机器的缺省 IP 地址，如图 8-3 所示。

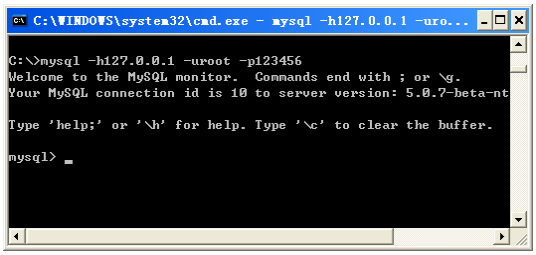


图 8-3 连接远程 MySQL 数据库

当然也可以直接通过 MySQL 的客户端登录。可以通过菜单【开始→所有程序→MySQL→MySQL Server 5.0→MySQL Command Line Client】进入到 MySQL 的客户端，如图 8-4 所示。此时系统默认以 root 用户登录，然后提示输入密码，密码输入正确表示登录成功。

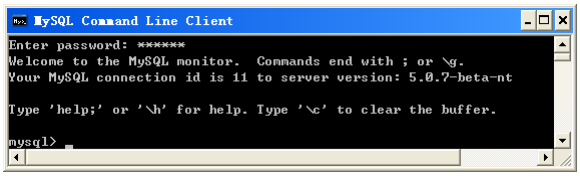
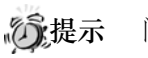


图 8-4 MySQL 客户端

8.1.2 修改用户密码

修改密码非常简单，只需要使用 mysqladmin 命令，格式如下：

```
mysqladmin -u用户名 -p旧密码 password 新密码
```



password 和新密码之间的空格不能省略，而-u 和用户名以及-p 和旧密码之间的空格可以省略。另外 mysqladmin 命令必须在命令行模式下执行，而不能在 MySQL 客户端模式下执行。

如果在修改密码的时候没有输入原始密码，而是输入以下格式：

```
mysqladmin -u用户名 -p password 新密码
```

则系统则会提示输入原始密码。图 8-5 所示演示了两种语句修改密码的过程。

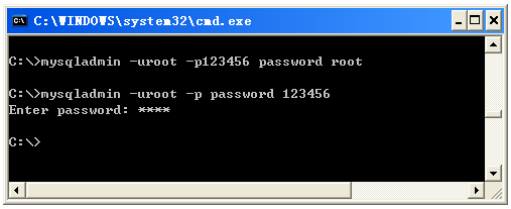


图 8-5　修改密码

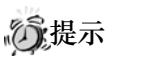
图 8-5 中，第 1 行是将原来的密码“123456”改成“root”，而第 2 行是将密码改回“123456”。因为第 2 行没有输入原始密码，则提示输入原始密码，因为第 1 行已经将密码修改为“root”，所以第 3 行应该输入密码“root”。

有了以上最基本的操作之后，接下来可以进行各种 MySQL 的管理工作了。

8.1.3　MySQL 的权限管理

MySQL 安装完之后，一般只有一个管理员用户，即“root”，该用户拥有最高权限，以便对数据库进行管理。而对于不同的开发人员应该分配不同的账号，同时为了安全起见，应该为这些用户设置密码。在为其他人员创建账号的同时，可以针对不同的账号设置不同的权限。在设置权限的时候，应该按照最少权限原则，即在能够满足用户需求的情况下，尽量少的分配其权限。

下面简单介绍 MySQL 的权限类型和级别。在 MySQL 中，通常有 3 种基本的权限：一般用户权限、管理员权限以及特定的权限。任何用户都可以被赋予这 3 种权限，但是应该给用户赋予最少权限，以满足需求为目的。



用户的权限被保存在一个名为“mysql”的数据库中。该数据库中包含了一系列的表来存储用户权限。

权限信息一般存在 MySQL 数据库的 user、db、host、tables_priv 和 columns_priv 表中。通常赋予一般开发人员的权限如表 8-1 所示。

表 8-1　普通用户权限

权　限	应用范围	说　明	权　限	应用范围	说　明
select	表、列	允许从表中选择记录	index	表	允许用户创建或者删除表索引
insert	表、列	允许向表中插入新记录	alter	表	允许用户改变表结构
update	表、列	允许用户修改表中的记录	create	数据库、表	允许创建新数据库或者表
delete	表	允许删除表中的记录	drop	数据库、表	允许用户删除数据库或者表

以上权限适合一般开发人员，获取这些权限之后，即使有恶意，一般也不会损害整个数据库服务器的运行。

表 8-2 所示列出了适用于管理员的权限。具备这些权限之后，就能够管理整个数据库系统。

表 8-2 管理员权限

权　限	说　明
CREATE TEMPORARY TABLES	允许创建临时表
FILE	允许从文件读入表，以及从表中读出文件
LOCK TABLES	允许使用 LOCK TABLE 语句用来锁定数据库表
PROCESS	允许查看所有用户的服务器进程
RELOAD	允许重新载入授权表、清空授权、主机、日志和表格
REPLICATION CLIENT	允许复制从服务器连接到主服务器
REPLICATION SLAVE	允许使用 SHOW DATABASES 语句查看所有数据库列表。如果没有这个权限，则只能查看权限规定的数据库
SHUTDOWN	允许关闭 MySQL 服务器
SUPER	允许关闭属于任何用户的线程

提示 在进行表 8-2 所示的权限授权时，要非常小心。

除了表 8-1 以及表 8-2 所示的权限之外，还有两个特别的权限，如表 8-3 所示。

表 8-3 特殊权限

权　限	说　明	权　限	说　明
ALL	允许创建临时表	USAGE	允许从文件读入表，以及从表中读出文件

8.1.4 创建新用户

前面介绍了 MySQL 系统的基本权限，接下来通过使用命令创建新的用户，以及为用户设置新的权限。

1．GRANT 命令

GRANT 命令用于创建新的用户并且赋予他一定的权限。在 MySQL 中，权限一般分成 4 个级别：全局、数据库、表和列。

在稍后章节中，会详细介绍这些权限。GRANT 命令的语法形式如下：

```
GRANT priv_type [(column_list)] [, priv_type [(column_list)] ...]
   ON {tbl_name | * | *.* | db_name.*}
   TO user_name [IDENTIFIED BY [PASSWORD] 'password']
      [, user_name [IDENTIFIED BY 'password'] ...]
   [REQUIRE
      NONE |
    [{SSL| X509}]
    [CIPHER cipher [AND]]
    [ISSUER issuer [AND]]
    [SUBJECT subject]]
   [WITH [GRANT OPTION | MAX_QUERIES_PER_HOUR # |
                    MAX_UPDATES_PER_HOUR # |
                    MAX_CONNECTIONS_PER_HOUR #]]
```

在以上表达式中，方括号内的选项表示是可选的。其中参数 priv_type 是由逗号分开的一组权限。参数 column_list 是可选的。可以用来对数据库表的每一列指定权限，也可以使用单列的名称或者用逗号分开的一组列的名称。tbl_name 表示被授权的数据库表的名

称，可以将其指定为“*.*”，这表示将权限应用到所有的数据库。这种授权方式称为全局权限。但是一般只有在对管理员授权时才会采取这种方法。如果要授权某个数据库中所有的表，则可以使用 dbname.*的形式，其中 dbname 是某个具体数据库的名称。如果要授权某个具体的数据库表，可以使用 dbname.tablename，其中 tablename 就是某个具体的表的名称。

参数 user_name 表示要被授权的用户名。用户名不必与登录系统时使用的名称一样。另外 MySQL 系统允许用户名包含一个主机名，这样可以区分来自不同域的用户，如 amipig@localhost、amipig@sina.com 等。这样可以允许来自不同域的用户使用相同的用户名，同时也可以指定用户从什么地方连接到本机，甚至可以指定他们在特定的时段可以访问哪些表或者数据库。

参数 password 表示用户在登录的时候应该使用的密码。在授权的时候，尽量使用密码。

REQUIRE 语句允许指定用户是否通过加密套接字连接，或者指定其他的 SSL 选项。

WITH GRANT OPTION 选项如果被指定，则表示允许指定的用户向别人授权自己拥有的权限。

参数 MAX_QUERIES_PER_HOUR #指定用户每小时能够执行的最大查询次数。

参数 MAX_UPDATES_PER_HOUR #指定用户每小时能够执行的最大更新次数。

参数 MAX_CONNECTIONS_PER_HOUR #指定用户每小时执行的最大连接数量。

下面通过实例来学习 GRANT 命令。

（1）授权 1，如下所示：

```
mysql> GRANT ALL  ON *.* TO monty@localhost
       IDENTIFIED BY 'abc' WITH GRANT OPTION;
```

以上代码授予 monty@localhost 用户所有的权限。其中用户密码为 abc。同时允许该用户向其他用户授权这些权限。



登录到 MySQL 客户端之后，在 mysql>提示符下面输入的所有命令必须以分号结束，如果不以分号结束，则程序会认为命令没有结束，不会执行该命令。

（2）授权 2，如下所示：

```
mysql> GRANT RELOAD,PROCESS ON *.* TO admin@localhost;
```

执行完以上语句之后，admin 用户可以从 localhost 进行一个不需要口令的连接并且被授予 reload 和 process 管理权限的用户。

（3）授权 3，如下所示：

```
mysql> GRANT USAGE ON *.* TO dummy@localhost;
```

以上语句创建了一个不具备任何权限的用户 dummy，该用户可以登录，但是不能进行任何操作。管理员可以在以后赋予其更多的权限。

以上 3 条授权语句的执行结果如图 8-6 所示。

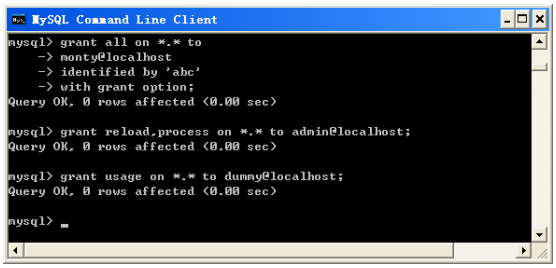


图 8-6　GRANT 命令的应用

提示 （1）如果输入命令时，回车后发现忘记加分号，无需重新输入一遍命令，只要打个分号回车就可以了。也就是说可以把一个完整的命令分成几行来打，最后用分号作为结束标志。（2）可以使用光标上下键调出以前的命令。

另外，如果对 MySQL 数据库管理有深入了解，还可以直接使用 SQL 语句向数据库中添加新用户以及权限。如下面的语句分别授予不同的用户权限：

```
mysql> INSERT INTO user VALUES('localhost','monty',PASSWORD('something'),
             'Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y')
mysql> INSERT INTO user VALUES('%','monty',PASSWORD('something'),
             'Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y')
mysql> INSERT INTO user SET Host='localhost',User='admin',
              Reload_priv='Y', Process_priv='Y';
mysql> INSERT INTO user (Host,User,Password)
                    VALUES('localhost','dummy','');
mysql> FLUSH PRIVILEGES;
```

2．REVOKE 命令

既然有授权新用户的命令，那肯定有注销某个用户的命令。REVOKE 命令用于收回用户权限。该命令的使用格式同 GRANT 基本类似，如下所示：

```
REVOKE priv_type [(column_list)] [, priv_type [(column_list)] ...]
   ON {tbl_name | * | *.* | db_name.*}
   FROM user_name [, user_name ...]
```

REVOKE 命令后面的参数与 GRANT 命令后面的参数含义一样。

3．查看所有用户

创建了一系列的用户后，管理员可以查看所有的用户。要查看所有的用户非常简单，只需要输入以下两个语句即可：

```
mysql>use mysql;
mysql>select * from user;
```

以上第一句表示使用系统管理数据库 mysql，第二句表示从 user 表中查询出所有的用户。

8.1.5 创建数据库

在 MySQL 数据库系统中，大部分标准的 SQL 语句都能够得到很好的支持。要创建一个数据库，使用的命令为 Create Database，该命令的格式如下：

```
mysql>CREATE DATABASE db_name;
```

其中 db_name 表示被创建的数据库名称。数据库名称的最大长度为 64。在给数据库名称命名的时候，不能含有“/”以及“.”等字符。

下面代码将创建一个名称为“bookstore”的数据库。

```
mysql>CREATE DATABASE bookstore;
```

创建完数据库之后，可以使用 DROP 命令来删除数据库，格式如下：

```
mysql>DROP DATABASE [IF EXISTS] db_name
```

DROP DATABASE 返回从数据库目录被删除的文件的数目。通常，3 倍于表的数量，因为每张表对应于一个“.MYD”文件、一个“.MYI”文件和一个“.frm”文件。如下面的代码将删除刚才创建的“bookstore”数据库。

```
mysql>DROP DATABASE bookstore;
```

以上命令的执行过程如图 8-7 所示。

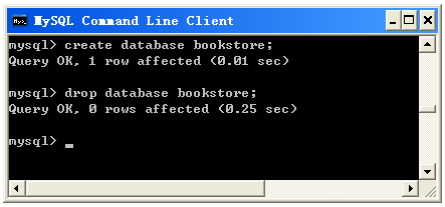


图 8-7　创建以及删除数据库

提示　　删除数据库使用命令“DROP DATABASE 数据库名称”。

8.1.6　创建数据库表

创建数据库之后，此时数据库是空的，没有任何数据库表。接下来介绍如何创建数据库表。创建数据库表用“Create Table”命令，该命令格式如下：

```
CREATE [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name [(create_definition,...)]
[table_options] [select_statement]
```

其中 tbl_name 为要创建的数据库表名。

create_definition 表示以下内容：

```
  col_name type [NOT NULL | NULL] [DEFAULT default_value] [AUTO_INCREMENT]
          [PRIMARY KEY] [reference_definition]
  or    PRIMARY KEY (index_col_name,...)
  or    KEY [index_name] (index_col_name,...)
  or    INDEX [index_name] (index_col_name,...)
  or    UNIQUE [INDEX] [index_name] (index_col_name,...)
  or    FULLTEXT [INDEX] [index_name] (index_col_name,...)
  or    [CONSTRAINT symbol] FOREIGN KEY [index_name] (index_col_name,...)
          [reference_definition]
  or    CHECK (expr)
```

其中 col_name 表示字段名，type 表示字段类型，而后面的内容表示字段的常用属性。如[NOT NULL | NULL]表示是否可以为空，[DEFAULT default_value]表示默认值，[AUTO_INCREMENT] 表示是否是自动编号，[PRIMARY KEY]表示是否为主键。

type 数据类型有很多，常用的数据类型如下（这些数据类型的含义参见下一节）：

```
       TINYINT[(length)] [UNSIGNED] [ZEROFILL]
  or    SMALLINT[(length)] [UNSIGNED] [ZEROFILL]
  or    MEDIUMINT[(length)] [UNSIGNED] [ZEROFILL]
  or    INT[(length)] [UNSIGNED] [ZEROFILL]
  or    INTEGER[(length)] [UNSIGNED] [ZEROFILL]
  or    BIGINT[(length)] [UNSIGNED] [ZEROFILL]
  or    REAL[(length,decimals)] [UNSIGNED] [ZEROFILL]
  or    DOUBLE[(length,decimals)] [UNSIGNED] [ZEROFILL]
  or    FLOAT[(length,decimals)] [UNSIGNED] [ZEROFILL]
  or    DECIMAL(length,decimals) [UNSIGNED] [ZEROFILL]
  or    NUMERIC(length,decimals) [UNSIGNED] [ZEROFILL]
  or    CHAR(length) [BINARY]
  or    VARCHAR(length) [BINARY]
  or    DATE
  or    TIME
  or    TIMESTAMP
```

```
  or    DATETIME
  or    TINYBLOB
  or    BLOB
  or    MEDIUMBLOB
  or    LONGBLOB
  or    TINYTEXT
  or    TEXT
  or    MEDIUMTEXT
  or    LONGTEXT
  or    ENUM(value1,value2,value3,...)
  or    SET(value1,value2,value3,...)
```

reference_definition 的定义如下：

```
REFERENCES tbl_name [(index_col_name,...)]
    [MATCH FULL | MATCH PARTIAL]
   [ON DELETE reference_option]
   [ON UPDATE reference_option]
```

reference_option 的定义如下：

```
    RESTRICT | CASCADE | SET NULL | NO ACTION | SET DEFAULT
```

table_options 的定义如下：

```
    TYPE = {BDB | HEAP | ISAM | InnoDB | MERGE | MRG_MYISAM | MYISAM }
or  AUTO_INCREMENT = #
or  AVG_ROW_LENGTH = #
or  CHECKSUM = {0 | 1}
or  COMMENT = "string"
or  MAX_ROWS = #
or  MIN_ROWS = #
or  PACK_KEYS = {0 | 1 | DEFAULT}
or  PASSWORD = "string"
or  DELAY_KEY_WRITE = {0 | 1}
or     ROW_FORMAT= { default | dynamic | fixed | compressed }
or  RAID_TYPE= {1 | STRIPED | RAID0 } RAID_CHUNKS=#  RAID_CHUNKSIZE=#
or  UNION = (table_name,[table_name...])
or  INSERT_METHOD= {NO | FIRST | LAST }
or     DATA DIRECTORY="absolute path to directory"
or     INDEX DIRECTORY="absolute path to directory"
```

select_statement 的定义如下：

```
    [IGNORE | REPLACE] SELECT ...  (Some legal select statement)
```

下面对以上的一些关键字进行文字解释。

（1）如果既没有指定 NULL 也没有指定 NOT NULL，则被视为指定了 NULL 。

（2）一个整型列可以有附加属性 AUTO_INCREMENT。当插入一个 NULL 值（推荐）或 0 到一个 AUTO_INCREMENT 列，该列将被设置到 value+1。在这里，value 是表中当前列的最大值。AUTO_INCREMENT 序列以 1 开始。如果以 AUTOCOMMIT 模式执行 DELETE FROM table_name（没有一个 WHERE 子句）删除表中的所有记录行，对于所有的表序列均重新开始。



注意 每个表只能有一个 AUTO_INCREMENT 列，并且必须被索引。

（3）DEFAULT 值必须是一个常量，不可以是一个函数或一个表达式。如果一个列没有

指定 DEFAULT 值，MySQL 将自动地赋予一个，规则如下：如果列可以接受 NULL 作为一个值，缺省值为 NULL。如果列被定义为 NOT NULL，缺省值取决于列的类型。

① 对于没有声明 AUTO_INCREMENT 属性的数字类型，缺省值为 0。对于一个 AUTO_INCREMENT 列，缺省值为序列中的下一个值。

② 对于非 TIMESTAMP 的日期和时间类型，缺省值是该类型适当的零值。对于表中的第一个 TIMESTAMP 列，缺省值为当前的日期和时间。

③ 对于非 ENUM 的字符串类型，缺省值是空字符串。对于 ENUM，缺省值为第一个枚举值。

缺省值必须是常量。这意味着，对于一个日期列，不能将一个像 NOW()或 CURRENT_DATE 的函数设置为缺省值。

（4）PRIMARY KEY 是一个惟一 KEY，它还有一个额外的约束，所有键列必须被定义为 NOT NULL。在 MySQL 中，该列被命名为 PRIMARY。一张表只能有一个 PRIMARY KEY。如果在表中没有一个 PRIMARY KEY，而某些应用程序要 PRIMARY KEY，MySQL 将返回第一个没有任何 NULL 列的 UNIQUE 键，做为 PRIMARY KEY。一个 PRIMARY KEY 可以是一个多列索引。然而，不能在一个列规格说明中使用 PRIMARY KEY 键属性来创建一个多列索引。这样做将仅仅标记单个列做为主键。必须使用 PRIMARY KEY（index_col_name, ...）句法。如果 PRIMARY 或 UNIQUE 键只由一个列组成，并且列类型是整型，则可以用_rowid 引用它。

虽然创建表格的命令看起来很复杂，但是平常使用的时候没这么复杂，其简化格式如下：

```
CREATE TABLE table_name(列名 1 属性,列名 2 属性…..)
```

提示　如果当前数据库表已经存在，则会产生错误。

当一个表被创建以后，系统会自动创建 3 个文件。

tbl_name.frm：表定义文件。

tbl_name.MYD：数据文件。

tbl_name.MYI：索引文件。

提示　删除表命令为 DROP TABLE，修改表的命令为 ALTER TABLE。

有了以上的一些基础知识，来看如何创建数据库表。因为 MySQL 一般是命令行模式，当输入大量的 SQL 语句的时候不是很方便，因此可以将 SQL 语句保存成一个.sql 文件，然后通过 MySQL 调用外部文件来执行这些 SQL 语句。为了方便说明，本节将创建一个在线书店的数据库（该数据库会在本书后面的在线书店系统中被使用）。

在本书对应章节下面有一个 bookstore_table.sql 文件，该文件中记录了在线书店需要的所有数据库表创建语句。该文件代码如下：

```
DROP TABLE IF EXISTS card_types;
DROP TABLE IF EXISTS categories;
DROP TABLE IF EXISTS editorial_categories;
DROP TABLE IF EXISTS editorials;
DROP TABLE IF EXISTS items;
DROP TABLE IF EXISTS members;
DROP TABLE IF EXISTS orders;
```

```
CREATE TABLE card_types (
      card_type_id       int auto_increment primary key,
      name              varchar(50) NOT NULL
);
CREATE TABLE categories (
      category_id        int  auto_increment primary key,
      name              varchar(50) NOT NULL
);
CREATE TABLE editorial_categories (
      editorial_cat_id    int  auto_increment primary key,
      editorial_cat_name   varchar(50) NULL
);
CREATE TABLE editorials (
      article_id         int auto_increment primary key,
      editorial_cat_id    int NULL DEFAULT 0,
      article_title       varchar(200) NULL,
      article_desc        text NULL,
      item_id            int NULL DEFAULT 0
);
CREATE TABLE items (
      item_id            int auto_increment primary key,
      category_id         int NOT NULL,
      name               varchar(255) NOT NULL,
      author             varchar(100) NULL,
      price              float NOT NULL,
      product_url         varchar(255) NULL,
      image_url          varchar(100) NULL,
      notes              text NULL,
      is_recommended      tinyint NULL DEFAULT 0,
      rating             int default 0,
      rating_count        int default 0
);
CREATE TABLE members (
      member_id          int auto_increment primary key,
      member_login        varchar(20) NOT NULL,
      member_password     varchar(20) NOT NULL,
      member_level        int NOT NULL DEFAULT 1,
      first_name         varchar(50) NOT NULL,
      last_name          varchar(50) NOT NULL,
      email              varchar(50) NOT NULL,
      phone              varchar(50) NULL,
      address            varchar(50) NULL,
      notes              text NULL,
      card_type_id        int NULL,
      card_number         varchar(50) NULL
);
CREATE TABLE orders (
      order_id           int auto_increment primary key,
      member_id          int NOT NULL,
      item_id            int NOT NULL,
      quantity           int NOT NULL
);
```

以上代码首先判断这些表是否存在，如果存在，则删除原来的表，然后重新创建。关于以上表的含义，将在本书后面的在线书店一章中介绍。

接下来在 MySQL 中执行该文件，调用外部文件的格式如下：

```
mysql>source sql 文件
```

调用 bookstore_table.sql 文件，可以采用以下命令（注意要先创建 bookstore 数据库）：

```
mysql>source d:\masterphp\chap08\code\bookstore_table.sql;
```

运行以上语句后，将会自动创建所有的数据库表。

8.1.7　数据类型

在创建数据库表的时候，需要指定要创建的每个列的数据类型。因此本节将介绍 MySQL 支持的数据类型以及含义。

MySQL 支持大量的列类型，可以被分为 3 类：数字类型、日期和时间类型以及字符串（字符）类型。本章节首先给出可用类型的概述，并且总结各类型的存储需求，然后提供对各类型中的类型范畴更详细的描述。

1．数字类型

MySQL 支持所有的 ANSI/ISO SQL92 数字类型。这些类型包括准确数字的数据类型（NUMERIC、DECIMAL、INTEGER 和 SMALLINT），也包括近似数字的数据类型（FLOAT、REAL 和 DOUBLE PRECISION）。关键词 INT 是 INTEGER 的同义词，关键词 DEC 是 DECIMAL 的同义词。

NUMERIC 和 DECIMAL 类型被 MySQL 以同样的类型实现，这在 SQL92 标准中是允许的。它们用于保存对准确精度有重要要求的值，例如与金钱有关的数据。

数字类型总体分成整型和浮点型两类。对于整型数据，可以指定无符号类型。对于所有数字类型，也可以指定 ZEROFILL 属性。当显示 ZEROFIL 字段中的值时，空余部分用前导 0 来补充。如果将一个字段指定为 ZEROFILL，它将自动成为 UNSIGNED 数据类型。

整数类型如表 8-4 所示。浮点数类型如表 8-5 所示。

表 8-4　整数数据类型

类　型	取值范围	存储空间	说　明
TINYINT[(m)]	–127～128 或 0～255	1	最小整型定义
BIT			和 TINYINT 一样
BOOL			和 TINYINT 一样
SMALLINT[(M)]	–32768～32767 或 0～65535	2	小型整数
MEDIUMINT[(M)]	–8388608～8388607 或 0～16777215	3	中型整数
INT[(M)]	–2^31～2^31–1 或 0～2^32–1	4	一般整数
INTEGER[(M)]			同 INT
BIGINT[(M)]	–2^63～2^63–1 或 0～2^64–1	8	大型整数

表 8-5　浮点数据类型

类　型	取值范围	存储空间	说　明
FLOAT(精度)	取决于精度	可变	可用于指定单精度或者双精度浮点数
FLOAT[(M,D)]	+(–)1.175494351E–38 +(–)3.402823466E+38	4	单精度浮点数。等同于 FLOAT(4)，但是指定显示宽度和小数位数
DOUBLE[(M,D)]	+(–)1.7976931348623157E+308 +(–)2.2250738585072014E–308	8	双精度浮点数，等同于 FLOAT(8)，但是指定显示宽度和小数位数

续表

类　型	取值范围	存储空间	说　明
DOUBLE PRECISION[(M,D)]	同上		同上
REAL[(M,D)]	同上	3	同上
DECIMAL[(M[,D])]	可变	M+2	一般整数
NUMERIC[(M,D)]	同上		DECIMAL 同义词
INTEGER[(M)]	同上		DECIMAL 同义词
DEC[(M,D)]	同上		DECIMAL 同义词

2．日期和时间型

日期和时间类型有 DATETIME、DATE、TIMESTAMP、TIME 和 YEAR。每一个类型均有合法值的范围，当给它们赋于一个不合法的值时，将会被“零”代替。


注意

MySQL 允许存储某个“不严格的”合法的日期，例如 1999-11-31。为了“加快”对日期的检查，MySQL 仅检查月份应在 0～12 范围内，以及天在 0～31 范围内。当一个应用程序希望存储一个出生日期，而并不知准确的出生日月时，这将变得非常有用。在这种情况下，可以简单地以 1999-00-00 或 1999-01-00 形式存储日期。

常用的日期和时间数据类型如表 8-6 所示。

表 8-6　　日期和时间数据类型

类　型	取值范围	说　明
DATE	1000-01-01 9999-12-31	日期，以 YYYY-MM-DD 格式显示
TIME	-838:58:59 838:59:59	时间，以 HH:MM:SS 格式显示
DATETIME	1000-01-01 00:00:00 9999-12-31 23:59:59	日期和时间，以 YYYY-MM-DD HH:MM:SS 格式显示
TIMESTAMP[(M)]	1970-01-01 00:00:00 2037 年的某个时间	时间标签，在处理报告时使用显示格式取决于 M 的值，见表 8-7
YEAR[(1\|4)]	70-69(1970-2069) 1901-2155	年份可以指定两位数字和四位数字的格式

表 8-7 所示显示了 TIMESTAMP 所有的可显示类型。

表 8-7　　TIMESTAMP 的显示类型

指定的类型	显　示	指定的类型	显　示
TIMESTAMP	YYYYMMDDHHMMSS	TIMESTAMP(8)	YYYYMMDD
TIMESTAMP(2)	YY	TIMESTAMP(10)	YYMMDDHHMM
TIMESTAMP(4)	YYMM	TIMESTAMP(12)	YYMMDDHHMMSS
TIMESTAMP(6)	YYMMDD	TIMESTAMP(14)	YYYYMMDDHHMMSS

下面介绍在使用日期时间类型时的注意事项。

（1）2000 年问题。MySQL 自身是 2000 年安全的，但是提交给 MySQL 的值可能并不是。任何一个包含 2 位年份的值是存在二义性的，因为世纪值是未知的。这样的值必须被解释为 4 位格式，因为 MySQL 内部使用 4 位存储年份。

对于 DATETIME、DATE、TIMESTAMP 和 YEAR 类型，MySQL 使用下列规则解释存在二义性的年份值：在范围 00～69 内的年值被转换为 2000～2069；在范围 70～99 内的年值被

转换为 1970～1999。


注意　某些像 MIN()和 MAX()之类的函数会将 TIMESTAMP/DATE 转换为一个数字。这就意味着，一个 2 位年份的时间戳将完全不能与这些函数一同工作。在这种情况下，解决的办法是将 TIMESTAMP/DATE 转换为 4 位年份格式或者使用诸如 MIN(DATE_ADD(timestamp,INTERVAL 0 DAYS))的方法。

（2）TIMESTAMP 列类型提供了一种类型，通过它可以以当前操作的日期和时间自动地标记 INSERT 或 UPDATE 操作。如果一张表中有多个 TIMESTAMP 列，只有第一个被自动更新。

自动更新第一个 TIMESTAMP 列在下列条件下进行：

① 列值没有明确地在一个 INSERT 或 LOAD DATA INFILE 语句中被指定。

② 列值没有明确地在一个 UPDATE 语句中被指定，并且其他的一些列值已发生改变。


注意　当一个 UPDATE 设置一个列值为它原有值时，将不会引起 TIMESTAMP 列的更新，因为，如果设置一个列值为它当前值时，MySQL 为了效率为忽略更新。

③ 明确地以 NULL 设置 TIMESTAMP 列。

第一个列以外其他 TIMESTAMP 列，可以设置到当前的日期和时间，只要将该列赋值 NULL 或 NOW()。

任何 TIMESTAMP 列均可以被设置一个不同于当前操作日期与时间的值，这通过为该列明确指定一个所期望的值来实现。这也适用于第一个 TIMESTAMP 列。这个选择性是很有用的，举例来说，当希望 TIMESTAMP 列保存该记录行被新添加时的当前的日期和时间，但该值不再发生改变，无论以后是否对该记录行进行更新。

当该记录行被建立时，让 MySQL 设置该列值。这将初始化该列为当前日期和时间。以后当对该记录行的其他列执行更新时，为 TIMESTAMP 列值明确地指定为它原来的值。

另一方面，可能发现更容易的方法，使用 DATETIME 列，当新建记录行时以 NOW()初始化该列，以后在对该记录行进行更新时不再处理它。

3．字符类型

字符类型通常分成 3 类。

（1）普通文本字符串，即小段文本。包括 CHAR（固定长度）类和 VARCHAR（可变长度字符）类型。

一个 CHAR 列的长度被固定为创建表所声明的长度。长度可以是 1～255 之间的任一值。当 CHAR 值被存储时，被用空格在右边填补到指定长度。当 CHAR 值被检索时，尾部的空格被截除。

VARCHAR 列的值是变长的字符串。可以声明一个 VARCHAR 列在 1～255，就像对 CHAR 列一样。然而，与 CHAR 相反的，VARCHAR 值只以所需的字符数存储，另加一个字节存储记录的长度。值并不被填补；相反的，当被存储时，尾部的空格被截除。

表 8-8 所示为常规字符串类型。

表 8-8　常规字符串类型

类　型	取 值 范 围	说　明
[NATIONAL] CHAR(M) [BINARY\|ASCII\|UNICODE]	0～255 个字符	固定长度为 M 的字符串，其中 M 的取值范围为 0～255。NATIONAL 关键字指定了应该使用的默认字符集。BINARY 关键字指定了数据是否区分大小写（默认就是区分大小写的）。ASCII 关键字指定了在改列中使用 latin1 字符集。UNICODE 关键字指定了使用 UCS 字符集

续表

类　　型	取 值 范 围	说　　明
CHAR		和 CHAR(1)相同
[NATIONAL]VARCHAR(M) [BINARY]	0～255 个字符	长度可变，其他同 CHAR(M)一样

表 8-9 所示通过在 CHAR(4)和 VARCHAR(4)列中存储不同的字符串，显示了两种类型列的不同。

表 8-9　　普通文本字符串类型

值	CHAR(4)	存 储 需 求	VARCHAR(4)	存 储 需 求
''	' '	4 字节	''	1 字节
'ab'	'ab '	4 字节	'ab'	3 字节
'abcd'	'abcd'	4 字节	'abcd'	5 字节
'abcdefgh'	'abcd'	4 字节	'abcd'	5 字节

CHAR 和 VARCHAR 列值以省略字母大小写的方式进行排序和比较，除非在表建立时 BINARY 属性被指定。BINARY 属性意味着，该列值根据 MySQL 服务器正在运行的机器的 ASCII 表顺序进行字母大小写敏感的方式排序和比较。BINARY 并不影响该列如何被存储和检索。

（2）TEXT 和 BLOB 类型。这些类型大小可变，其中 TEXT 适合存储长文本，而 BLOB 适合存储二进制数据，它支持任何数据，如文本、视频、声音和图像等。

表 8-10 所示给出了 TEXT 和 BLOB 类型。以字符计算的 TEXT 字段最大长度是可以存储在该字段中文件的最大字节数。

表 8-10　　TEXT 和 BLOB 类型

类　　型	最大长度（字节数）	说　　明
TINYBLOB	2^8－1(255)	小 BLOB 字段
TINYTEXT	2^8－1(255)	小 TEXT 字段
BLOB	2^16－1(65 535)	常规 BLOB 字段
TEXT	2^16－1(65 535)	常规 TEXT 字段
MEDIUMBLOB	2^24－1(16 777 215)	中型 BLOB 字段
MEDIUMTEXT	2^24－1(16 777 215)	中型 TEXT 字段
LONGBLOB	2^32－1(4 294 967 295)	长 BLOB 字段
LONGTEXT	2^32－1(4 294 967 295)	长 TEXT 字段

一个BLOB是一个可以保存可变数量的数据的二进制对象。4个BLOB类型（TINYBLOB、BLOB、MEDIUMBLOB 和 LONGBLOB）之间的不同仅在于它们能保存值的最大长度不一致。

4 个 TEXT 类型（TINYTEXT、TEXT、MEDIUMTEXT 和 LONGTEXT）对应于 4 个 BLOB 类型，并且有相同的最大长度和存储需求。在 BLOB 和 TEXT 类型之间的惟一差别就是，对 BLOB 值是以字母大小写敏感的方式进行排序和比较的，而对 TEXT 值以忽略字母大小写方式进行排序和比较。换句话说，TEXT 是一个忽略字母大小写的 BLOB。

如果将一个超过列类型最大长度的值赋给 BLOB 或 TEXT 列，该值将被截断以适应它。

（3）两种特殊类型，SET 和 ENUM。

表 8-11 所示为 ENUM 和 SET 类型。

表 8-11　ENUM 和 SET 类型

类　型	集合中的最大值	说　明
ENUM('value1','value2',…)	65 535	该类型的列只可以容纳所列值之一或者为 NULL
SET('value1','value2',..)	64	该类型的列可以容纳一组值或者为 NULL

ENUM 是一个字符串对象，其值通常选自一个允许值列表中，该列表在表创建时的列规格说明中被明确地列举。

在下列某些情况下，ENUM 值也可以是空串（""）或 NULL。

（1）如果将一个无效值插入一个 ENUM（即一个不在允许值列表中的字符串），空字符串将作为一个特殊的错误值被插入。事实上，这个字符串有别于一个“普通的”空字符串，因为这个字符串有个数字索引值为 0。

（2）如果一个 ENUM 被声明为 NULL，NULL 也是该列的一个合法值，并且该列的缺省值也将为 NULL。如果一个 ENUM 被声明为 NOT NULL，该列的缺省值将是该列表所允许值的第一个成员。

每个枚举值均有一个索引值：在列说明中列表值所允许的成员值被从 1 开始编号。空字符串错误值的索引值为 0。这就意味着，可以使用下面所示的 SELECT 语句找出被赋于无效 ENUM 值的记录行。

```
mysql> SELECT * FROM tbl_name WHERE enum_col=0;
```

如果在一个数字语境中检索一个 NUM，列值的索引值将被返回。例如，可以像这样使用数字值检索一个 ENUM 列：

```
mysql> SELECT enum_col+0 FROM tbl_name;
```

如果将一个数字存储到一个 ENUM 中，数字被作为一个索引值，并且存储的值是该索引值所对应的枚举成员。

SET 是一个字符串对象，它可以有 0 或更多个值，每个值均必须选自一个允许值列表中，该列表在表创建时被指定。包含多个集合成员的 SET 列值，由逗号（“,”）将各成员分隔。

8.1.8　查看数据库

创建完数据库以及数据库表之后，可以来查看数据库中表的信息。常见的命令有 show 和 describe。

1．show 命令

通过 show 命令可以查看数据库中所有的表，例如要查看 bookstore 数据库中的所有表，可以使用如下命令：

```
mysql> use bookstore;
Database changed
mysql> show tables;
+----------------------+
| Tables_in_bookstore  |
+----------------------+
| card_types           |
| categories           |
| editorial_categories |
| editorials           |
| items                |
| members              |
| orders               |
+----------------------+
7 rows in set (0.00 sec)
```

通过命令“show tables”可以查看所有的数据库表格。另外如果想查看系统中所有的数据库列表，可以使用“show databases”命令，如下所示：

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| bookstore          |
| mysql              |
| test               |
| test1              |
+--------------------+
5 rows in set (0.01 sec)
```

以上显示了 mysql 中已经创建的数据库。其中 test、test1 以及 bookstore 数据库为笔者自己创建的。

另外还可以使用“show tables form 数据库名称”来检索数据库中的表。

```
mysql> show tables from bookstore;
+----------------------+
| Tables_in_bookstore  |
+----------------------+
| card_types           |
| categories           |
| editorial_categories |
| editorials           |
| items                |
| members              |
| orders               |
+----------------------+
7 rows in set (0.00 sec)
```



提示 show 命令是一个功能非常强大的命令，通过该命令，能够完成很多任务，具体功能如表 8-12 所示（MySQL 数据库是大小写不敏感的）。

表 8-12 show 命令

show 命令	说　明
SHOW DATABASES[LIKE database]	列出所有可供使用的数据库，数据库名称是可选的
SHOW [OPEN] TABLES [FROM database][LIKE table]	列出当前数据库或者名为 database 数据库中的表，表名是可选的
SHOW COLUMNS FROM table [FROM database][LIKE column]	列出当前数据库或者指定数据库中指定表的所有列，列的名称是可选的。可以使用 SHOW FIELDS 代替 SHOW COLUMNS
SHOW INDEX FROM table [FROM database]	显示当前数据库或者指定数据库中所有索引的详细信息，也可以用 SHOW KEYS 来代替
SHOW STATUS [LIKE status_item]	给出关于一些系统特定资源的信息，例如，正在运行的线程数量。LIKE 子句用于匹配这些资源名
SHOW [GLOBAL\|SESSION] VARIABLES [LIKE variable_name]	显示 MySQL 系统变量的名称和值，如版本号等
SHOW [FULL] PROCESSLIST	显示当前系统中运行的所有进程
SHOW TABLE STATUS [FROM database][LIKE database]	显示当前或者指定数据库每个表的信息，该信息包括表的类型和每个表的最新更新时间
SHOW GRANT FOR user	显示授予在 user 指定的用户其当前级别的权限所要求的 GRANT 语句
SHOW PRIVILEGES	显示服务器所支持的不同权限
SHOW CREATE DATABASE db	显示 CREATE DATABASE 语句是否能够创建指定的数据库
SHOW CREATE TABLE tablename	显示 CREATE TABLE 语句是否能够创建指定的表
SHOW [STORAGE] ENGINES	显示安装以后可用的存储引擎和默认引擎

续表

SHOW 命令	说　明
SHOW INNODB STATUS	显示 InnoDB 存储引擎的当前状态
SHOW [BDB] LOGS	显示 DBD 存储引擎的日志
SHOW WARNINGS [LIMIT[offset,]row_count]	显示最后一个执行的语句所产生的错误、警告和通知
SHOW ERRORS [LIMIT [offset,]row_count]	显示最后一个执行语句所产生的错误

2．describe 命令

describe 命令用于查看某个特定表的详细设计信息，例如为了查看 bookstore 数据库中 items 表的设计信息，可以使用如下命令：

```
mysql> describe items;
+----------------+--------------+------+-----+---------+----------------+
| Field          | Type         | Null | Key | Default | Extra          |
+----------------+--------------+------+-----+---------+----------------+
| item_id        | int(11)      | NO   | PRI | NULL    | auto_increment |
| category_id    | int(11)      | NO   |     |         |                |
| name           | varchar(255) | NO   |     |         |                |
| author         | varchar(100) | YES  |     | NULL    |                |
| price          | float        | NO   |     |         |                |
| product_url    | varchar(255) | YES  |     | NULL    |                |
| image_url      | varchar(100) | YES  |     | NULL    |                |
| notes          | text         | YES  |     | NULL    |                |
| is_recommended | tinyint(4)   | YES  |     | 0       |                |
| rating         | int(11)      | YES  |     | 0       |                |
| rating_count   | int(11)      | YES  |     | 0       |                |
+----------------+--------------+------+-----+---------+----------------+
11 rows in set (0.00 sec)
```

以上显示了 items 表的设计结构。

另外还可以通过“show comuns”命令来查看数据库表的列名，可以有两种使用方式：

```
show columns form 表名 from 数据库名
```

或者：

```
show columns form 数据库名.表名
```

具体应用如下。

```
mysql> show columns from bookstore.items;
+----------------+--------------+------+-----+---------+----------------+
| Field          | Type         | Null | Key | Default | Extra          |
+----------------+--------------+------+-----+---------+----------------+
| item_id        | int(11)      | NO   | PRI | NULL    | auto_increment |
| category_id    | int(11)      | NO   |     |         |                |
| name           | varchar(255) | NO   |     |         |                |
| author         | varchar(100) | YES  |     | NULL    |                |
| price          | float        | NO   |     |         |                |
| product_url    | varchar(255) | YES  |     | NULL    |                |
| image_url      | varchar(100) | YES  |     | NULL    |                |
| notes          | text         | YES  |     | NULL    |                |
| is_recommended | tinyint(4)   | YES  |     | 0       |                |
| rating         | int(11)      | YES  |     | 0       |                |
| rating_count   | int(11)      | YES  |     | 0       |                |
+----------------+--------------+------+-----+---------+----------------+
11 rows in set (0.00 sec)
```

另外通过 describe 命令可以查询某个具体列的信息。

```
mysql> describe items item_id;
+---------+---------+------+-----+---------+----------------+
| Field   | Type    | Null | Key | Default | Extra          |
+---------+---------+------+-----+---------+----------------+
| item_id | int(11) | NO   | PRI | NULL    | auto_increment |
+---------+---------+------+-----+---------+----------------+
1 row in set (0.11 sec)
```

其中 item_id 是 items 表的一个字段名称。

3．explain 命令

explain 命令类似于 show 和 descrebe 命令，例如可以通过 explain 命令来查询数据表的信息，如下所示：

```
mysql> explain items;
+----------------+--------------+------+-----+---------+----------------+
| Field          | Type         | Null | Key | Default | Extra          |
+----------------+--------------+------+-----+---------+----------------+
| item_id        | int(11)      | NO   | PRI | NULL    | auto_increment |
| category_id    | int(11)      | NO   |     |         |                |
| name           | varchar(255) | NO   |     |         |                |
| author         | varchar(100) | YES  |     | NULL    |                |
| price          | float        | NO   |     |         |                |
| product_url    | varchar(255) | YES  |     | NULL    |                |
| image_url      | varchar(100) | YES  |     | NULL    |                |
| notes          | text         | YES  |     | NULL    |                |
| is_recommended | tinyint(4)   | YES  |     | 0       |                |
| rating         | int(11)      | YES  |     | 0       |                |
| rating_count   | int(11)      | YES  |     | 0       |                |
+----------------+--------------+------+-----+---------+----------------+
11 rows in set (0.05 sec)

mysql> explain items price;
+-------+-------+------+-----+---------+-------+
| Field | Type  | Null | Key | Default | Extra |
+-------+-------+------+-----+---------+-------+
| price | float | NO   |     |         |       |
+-------+-------+------+-----+---------+-------+
1 row in set (0.00 sec)
```

提示 explain 命令还能够监视查询语句的执行过程。

8.1.9 创建索引

很多初学者对索引并不在乎，如果要进入商业程序的开发，索引就显得非常重要。索引在查询、统计的时候，能够大大地提高系统的计算速度。笔者曾经做过试验，对于同一个数据统计，如果设定索引，速度可以提高 10 多倍。如果要经常针对某个列进行查询和统计，可以为该列创建索引。

创建索引使用的命令为“CREATE INDEX”，该命令的使用格式如下：

```
CREATE [UNIQUE|FULLTEXT] INDEX index_name
      ON tbl_name (col_name[(length)],...
```

通常在创建数据库表格的时候可以直接创建索引，如果创建表格后，可以通过 CREATE INDEX 重新创建索引。其中 FULLTEXT 索引用来索引文本字段。可选的 length 字段允许指定只有该字段的前 length 个字符被索引。index_name 是索引名称，tbl_name 是表名称，而 col_name 是列名称。

8.1.10 备份数据库

备份数据库是数据库管理员必须掌握的知识，因为只有这样才能确保当数据库受到破坏的时候，能够恢复原来的数据。

在 MySQL 中，备份数据库有很多种方法，下面介绍常用的几种方法。

（1）通过 LOCK TABLES 命令，可以进行数据库的备份，其格式如下：

```
LOCK TABLES tbl_name [AS alias] {READ [LOCAL] | [LOW_PRIORITY] WRITE}
          [, tbl_name [AS alias] {READ [LOCAL] | [LOW_PRIORITY] WRITE} ...]
```

其中 tbl_name 是要被锁定的数据库表名，可以同时锁定多个表格。而 READ 和 WRITE 为锁定的类型。如果用于备份，只需要使用 READ 锁定方式即可。同时在执行备份之前，必须要执行“FLUSH TABLES”命令来确保对索引所做的任何修改将写入到磁盘。但是使用锁定表方式来备份并不常用。

（2）mysql_dump 命令能够将整个数据库以 SQL 语句的方式导出到一个.sql 文本文件中。保存在日志文件中的更新将给出上次备份后数据库发生的变化。为了比较完整的备份数据库，最好备份所有的日志文件。

mysql_dump 命令的使用格式如下：

```
mysqldump [OPTIONS] database [tables]
```

其中 database 为数据库名称，tables 为要备份的表名。如果不给定任何表，整个数据库将被备份。通过执行“mysqldump –help”命令，能得到 mysqldump 版本支持的选项表。



如果运行 mysqldump 没有--quick 或--opt 选项，mysqldump 会将整个结果先写到内存中，如果正在备份一个大的数据库，这可能是一个问题。

mysqldump 支持下列选项。

--add-locks：在每个表备份之前增加 LOCK TABLES 并且之后 UNLOCK TABLE。

--add-drop-table：在每个 create 语句之前增加一个 drop table。

--allow-keywords：允许创建是关键词的列名字。由表名前缀于每个列名实现。

-c，--complete-insert：使用完整的 insert 语句（用列名字）。

-C，--compress：如果客户和服务器均支持压缩，压缩两者间所有的信息。

--delayed：用 INSERT DELAYED 命令插入行。

-e，--extended-insert：使用全新多行 INSERT 语法。

-#，--debug[=option_string]：跟踪程序的使用。

--help：显示一条帮助消息并且退出。

-F，--flush-logs：在开始备份前，去除在 MySQL 服务器中的日志文件。

-l，--lock-tables：为开始备份锁定所有表。

-t，--no-create-info：不写入表创建信息（CREATE TABLE 语句）。

-d，--no-data：不写入表的任何行信息。如果只想得到一个表的结构的备份，这是很有用的。

--opt：同--quick --add-drop-table --add-locks --extended-insert --lock-tables。

-pyour_pass，--password[=your_pass]：与服务器连接时使用的口令。如果不指定“=your_pass”部分，mysqldump 需要来自终端的口令。

-P port_num，--port=port_num：与一台主机连接时使用的 TCP/IP 端口号。

-q，--quick：不缓冲查询，直接备份至 stdout，使用 mysql_use_result()。

-T，--tab=path-to-some-directory：对于每个给定的表，创建一个 table_name.sql 文件，它包含 SQL CREATE 命令，和一个 table_name.txt 文件，包含数据。



这只有在 mysqldump 运行在 mysqld 守护进程运行的同一台机器上的时候才工作。.txt 文件的格式根据--fields-xxx 和--lines--xxx 选项来定。

-u user_name，--user=user_name：与服务器连接时，MySQL 使用的用户名。

-V，--version：打印版本信息并且退出。

最常见的 mysqldump 命令是制作整个数据库的一个备份：

```
mysqldump --opt database > backup.sql
```

如将前面创建的 bookstore 数据库备份，可以使用下面的命令：

```
C:\>mysqldump -uroot -p123456 --opt bookstore>d:\aa.sql
```

其中-u 和-p 表示连接的用户名和密码，而数据文件被备份到 aa.sql 文件中。

（3）直接复制数据库文件。该方法最直接和快速，直接将需要备份的数据库文件全部备份即可，只是在备份的时候要关闭 MySQL 服务器，确保没有用户在操作。

8.1.11 恢复数据库

恢复数据库就是当数据库受到破坏的时候，通过恢复数据库来修复。不同的备份方法，需要采用不同的恢复方式。

如果采用了第（1）种方法来备份数据库，可以直接将数据库文件重新复制到安装 MySQL 的相同位置。

如果采用第（2）种方法来备份数据库可以直接运行备份后的 SQL 文件，使得数据库恢复到备份时的状态。接着，还应该将数据库更新到保存在二进制日志文件中的状态。可以使用 mysqlbinlog 命令。

8.2 使用 MySQL 数据库

前面大致介绍了一些 MySQL 的管理知识，下面进入到具体学习阶段，为了方便学习，首先介绍在 MySQL 中如何进行代码注释。

MySQL 服务器支持“#到该行结束”、“--到该行结束”以及“/* 行中间或多个行*/” 的注释方式，如下所示：

```
mysql> SELECT 1+1;     # 这个注释直到该行结束
mysql> SELECT 1+1;     -- 这个注释直到该行结束
mysql> SELECT 1 /* 这是一个在行中间的注释 */ + 1;
mysql> SELECT 1+
/*
这是一个
多行注释的形式
*/
```

8.2.1 插入数据

通常在学习 SQL 语句的时候，总是先学习 SELECT 语句，但是由于本章创建的数据库中表中的记录都是空的，为了方便使用 SELECT 查询语句，要先学会如何向数据库插入记录。

在 MySQL 中，要插入数据，使用 INSERT 命令。INSERT 命令的格式如下所示：

```
  INSERT [LOW_PRIORITY | DELAYED] [IGNORE]
      [INTO] tbl_name [(col_name,...)]
      VALUES ((expression | DEFAULT),...),(...),...
      [ ON DUPLICATE KEY UPDATE col_name=expression, ... ]
```

或者：

```
  INSERT [LOW_PRIORITY | DELAYED] [IGNORE]
      [INTO] tbl_name [(col_name,...)]
      SELECT ...
```

或者：

```
INSERT [LOW_PRIORITY | DELAYED] [IGNORE]
     [INTO] tbl_name
     SET col_name=(expression | DEFAULT), ...
     [ ON DUPLICATE KEY UPDATE col_name=expression, ... ]
```

INSERT 将新行插入到一个已存在的表中。INSERT ... VALUES 形式的语句基于明确的值插入记录行。INSERT ... SELECT 形式的语句从另一个或多个表中选取出值，并将其插入。tbl_name 是记录将要被插入的表。

列名列表或 SET 子句指出语句指定的值赋给哪个列：如果在 INSERT ... VALUES 或 INSERT ... SELECT 中没有指定列列表，那么所有列的值必须在 VALUES()列表中或由 SELECT 提供。如果不知道表的列的次序，可以使用 DESCRIBE tbl_name 来决定。

任何没有明确指定一个值的列均会被设置为它的缺省值。举例来说，如果指定了一个列列表没有指定表中所有的列，未指定的列将被设置为它们的缺省值。

一个 expression 可以引用先前在值列表中设置的任何列。例如：

```
mysql> INSERT INTO tbl_name (col1,col2) VALUES(15,col1*2);
```

但是不能：

```
mysql> INSERT INTO tbl_name (col1,col2) VALUES(col2*2,15);
```

如果指定关键词 LOW_PRIORITY，INSERT 的执行将会被延迟，直到没有其他客户端正在读取表。在这种情况下，客户端不得不等待插入语句被完成，如果表被频繁地使用，那么这将会花费很长一段时间。这与 INSERT DELAYED 让客户端立即继续执行正好相反。

如果在一个有许多条记录行值的 INSERT 中指定关键词 IGNORE，任何在表中现有的 PRIMARY 或 UNIQUE 键上重复的记录行均会被忽略而不被插入。如果不指定 IGNORE，当有任何记录行在一个现有的键值上重复时，插入均会被中止。可以通过函数 mysql_info()测定共有多少记录行被插入到表中。

如果指定 ON DUPLICATE KEY UPDATE 子句，并且被插入的一个记录行在 PRIMARY 或 UNIQUE 键上将会产生一个重复值，那么记录行将被更新。举例来说：

```
mysql> INSERT INTO table (a,b,c) VALUES (1,2,3)
   --> ON DUPLICATE KEY UPDATE c=c+1;
```

假设列 a 被定义为 UNIQUE，并且已存在了一个 1，它将与下面的语句产生同样的结果：

```
mysql> UPDATE table SET c=c+1 WHERE a=1;
```

如果列 b 也是惟一的，UPDATE 命令将要被写成：

```
mysql> UPDATE table SET c=c+1 WHERE a=1 OR b=2 LIMIT 1;
```

并且如果 a=1 或者 b=2 匹配几个记录行，只有一个记录行将被更新。大体上，在有多重 UNIQUE 键的表上，应该尽量避免使用 ON DUPLICATE KEY 子句。当使用了 ON DUPLICATE KEY UPDATE 后，DELAYED 选项将被忽略。

如果 MySQL 被设置为使用 DONT_USE_DEFAULT_FIELDS 选项，INSERT 语句将产生一个错误，除非为所有需要一个非 NULL 值的列明确指定值。

通过使用 mysql_insert_id 函数可以找到用于一个 AUTO_INCREMENT 列的值。

如果使用 INSERT ... SELECT 或一个 INSERT ... VALUES 语句插入多值列，可以使用函数 mysql_info()得到查询的信息，格式如下：

```
Records: 100 Duplicates: 0 Warnings: 0
```

Duplicates 指出因与某些现有的惟一索引值重复而不能被插入的记录行数目。Warnings

指出在尝试插入的列值中，某些方面可能有问题的数目。在下列任何一个条件下，警告都会发生。

（1）向一个定义为 NOT NULL 的列中插入 NULL 值。该列被设置为它的缺省值。

（2）将一个超出列范围的值赋给一个数字列。该值被剪切到该范围内的适当的端点。

（3）将一个例如'10.34 a'的值赋给一个数字列。尾部的无用信息将被剥离，保留数字部分并将其插入。如果该值根本就不是一个数字，该列将被设置为 0。

（4）将一个超出了列最大长度的字符串插入到一个 CHAR、VARCHAR、TEXT 或 BLOB 列中。该值将被剪切到该列的最大长度。

（5）将一个对列类型不合法的值插入到一个日期或时间列中。

1．INSERT ... VALUES 语句

INSERT ... VALUES 是最常用的语句。下面将通过实例来介绍如何将记录添加到数据库中，这里主要演示向前面创建的 bookstore 数据库中的各个表中添加记录。

（1）向 categories 数据库表添加数据。

```
    insert into categories (name) values ('Programming');
    insert into categories (name) values ('Databases');
    insert into categories (name) values ('HTML & Web design');
```

以上向 categories 表中添加了 3 条记录，因为 categories 的第一个字段 category_id 是自动编号的，因此插入记录的时候只需要插入一个字段。

（2）向 items 数据库表添加记录。

```
    insert into items (category_id, name, author, price, product_url, image_url,
notes, is_recommended)
          values(2, 'Web Database Development：Step by Step', 'Jim Buyens', 39.99,
'http://www.amazon.com/exec/obidos/ASIN/0735609667/yessoftware',
'images/books/0735609667.jpg', 'As Web sites continue to grow in complexity and
in the volume of data they must present, databases increasingly drive their content.
WEB DATABASE DEVELOPMENT FUNDAMENTALS is ideal for the beginning-to-intermediate
Web developer, departmental power user, or entrepreneur who wants to step up to
a database-driven Web site-without buying several in-depth guides to the different
technologies involved. This book uses the clear Microsoft(r) Step by Step tutorial
method to familiarize developers with the technologies for building smart Web sites
that present data more easily. ', 1);
```

以上向 iterm 表添加了一条记录，用于保存图书的详细信息。

（3）向 members 表添加记录。

```
    insert into members (member_login, member_password, member_level, first_name,
last_name, email) values ('admin', 'admin', 2, 'Administrator', 'Account',
'admin@localhost');
    insert into members (member_login, member_password, member_level, first_name,
last_name, email) values ('guest', 'guest', 1, 'Guest', 'Account',
'guest@localhost');
```

由于手工添加记录比较烦琐，所以笔者已经将所有要插入记录的语句保存成 bookstore.sql 文件，读者只需要使用以下命令执行该文件即可。

```
    mysql>source d:\bookstore.sql;
```

提示 路径需按照实际设定修改。

2．INSERT ... SELECT 语句

使用 INSERT ... SELECT 语句，可以从一个或多个表中读取多个记录行，并将其快速地插入到一个表中，如下所示：

```
INSERT INTO tblTemp2 (fldID) SELECT tblTemp1.fldOrder_ID FROM tblTemp1 WHERE
tblTemp1.fldOrder_ID > 100;
```



INSERT 语句中的目标表不能在 SELECT 查询部分的 FROM 子句中出现，因为在 ANSI SQL 中，禁止从正在插入的表中选择。

8.2.2　查询数据

进行数据查询，使用 SELECT 关键字，其使用格式如下：

```
SELECT [STRAIGHT_JOIN]
      [SQL_SMALL_RESULT] [SQL_BIG_RESULT] [SQL_BUFFER_RESULT]
      [SQL_CACHE | SQL_NO_CACHE] [SQL_CALC_FOUND_ROWS] [HIGH_PRIORITY]
      [DISTINCT | DISTINCTROW | ALL]
   select_expression,...
   [INTO {OUTFILE | DUMPFILE} 'file_name' export_options]
   [FROM table_references
     [WHERE where_definition]
     [GROUP BY {unsigned_integer | col_name | formula} [ASC | DESC], ...]
     [HAVING where_definition]
     [ORDER BY {unsigned_integer | col_name | formula} [ASC | DESC] ,...]
     [LIMIT [offset,] rows | rows OFFSET offset]
     [PROCEDURE procedure_name(argument_list)]
     [FOR UPDATE | LOCK IN SHARE MODE]]
```

SELECT 用于检索从一个或多个表中选出的行。select_expression 表示希望检索的列。下面通过实例来介绍 SELECT 语句的应用。

（1）检索表中的全部记录。从单独一个表中检索全部记录是最简单的应用，如下所示：

```
mysql> select * from categories;
+-------------+-------------------+
| category_id | name              |
+-------------+-------------------+
|           1 | Programming       |
|           2 | Databases         |
|           3 | HTML & Web design |
+-------------+-------------------+
3 rows in set (0.02 sec)
```

以上语句从 categories 表中查询出所有记录。

（2）从表中获取指定列的数据。一个数据库表中可能含有很多的列，但是许多列是不需要的，此时可以指定需要检索和查询的列，如下所示：

```
mysql> select name from categories;
+-------------------+
| name              |
+-------------------+
| Programming       |
| Databases         |
| HTML & Web design |
+-------------------+
3 rows in set (0.09 sec)
```

以上只查询出列名为 name 的所有记录。用户可以指定多个列。多个列之间可以用逗号

隔开，如下所示：

```
mysql> select member_id,member_login,
    ->member_password from members ;
+-----------+--------------+-----------------+
| member_id | member_login | member_password |
+-----------+--------------+-----------------+
|         1 | admin        | admin           |
|         2 | guest        | guest           |
+-----------+--------------+-----------------+
2 rows in set (0.03 sec)
```

以上从 members 表中查询 member_id、member_login 以及 member_password 三个字段的所有记录。

（3）获取指定条件的记录。很多时候，需要显示的并不是所有的记录，尤其是在记录很多的时候。只有设定了查询条件，查询才有实际意义。比如要查找一个名为“张三”的人，此时需要对查询语句进行设定。设定查询语句，是通过 WHERE 子句来实现的。

最简单的查询条件即“=”（等于），用来查询完全符合设定条件的记录，如下所示：

```
mysql> select * from categories
    -> where name='Databases';
+-------------+-----------+
| category_id | name      |
+-------------+-----------+
|           2 | Databases |
+-------------+-----------+
1 row in set (0.00 sec)
```

以上是从数据库表 categories 表中查询出字段 name 值为“Databases”的记录，从执行结果可以看出已经找到了结果。

其实 Where 子句功能非常强大，用户可以进行更多项复杂的条件查询。表 8-13 所示列出了常用的 Where 子句的比较运算符。

表 8-13 常用的 Where 子句比较运算符

运 算 符	名 称	应 用	运 算 符	名 称	应 用
=	等于	会员编号=3	IS NULL	n/a	会员编号 IS NULL
>	大于	会员编号>3	BETWEEN	n/a	会员编号 BETWEEN 1 AND 10
<	小于	会员编号<3	IN	n/a	会员编号 IN (1,2,3,)
>=	大于等于	会员编号>=3	NOT IN	n/a	会员名称 NOT IN ('CAT','DOG')
<=	小于等于	会员编号<=3	LIKE	模式匹配	会员名称 LIKE ('jerry %')
!=或者<>	不等于	会员编号!=3	NOT LIKE	模式匹配	会员名称 NOT LIKE ('jerry %')
IS NOT NULL	n/a	会员编号 IS NOT NULL	REGEXP	常规表达式	会员名称正则表达式

以上介绍了 Where 子句常用的条件比较符号，下面通过实例来学习这些常用的比较子句。

如果要获取会员编号大于 1 的记录，如下所示：

```
mysql> select member_id,member_login
    -> from members where
    -> member_id>1;
+-----------+--------------+
| member_id | member_login |
+-----------+--------------+
|         2 | guest        |
+-----------+--------------+
1 row in set (0.00 sec)
```

以上查询出来一条结果，从结果看，因为 member_id 为 2，大于 1，所以记录被检索。

如果要获取会员编号为 1 或者为 3 的记录，如下所示：

```
mysql> select * from categories
    -> where category_id in (1,3);
+-------------+--------------------+
| category_id | name               |
+-------------+--------------------+
|           1 | Programming        |
|           3 | HTML & Web design  |
+-------------+--------------------+
2 rows in set (0.00 sec)
```

以上语句使用了 in 关键字来查询，当有很多个并列项的时候，使用该关键字非常方便，尤其在进行多个字符串检索的时候，如下所示：

```
mysql> select * from categories
    -> where name in
    -> ('Databases','Programming');
+-------------+-------------+
| category_id | name        |
+-------------+-------------+
|           1 | Programming |
|           2 | Databases   |
+-------------+-------------+
2 rows in set (0.00 sec)
```

如果要检索在一定范围内的记录，可以使用 between…and 语句，如下所示：

```
mysql> select * from categories
    -> where category_id between 1 and 5;
+-------------+--------------------+
| category_id | name               |
+-------------+--------------------+
|           1 | Programming        |
|           2 | Databases          |
|           3 | HTML & Web design  |
+-------------+--------------------+
3 rows in set (0.01 sec)
```

以上是查询 category_id 的值为 1 和 5 之间的记录。

很多时候，例如在网络书店买书，很可能用户不知道每本书的全名，而只知道其中的关键字，此时需要模糊查询。模糊查询的时候，使用关键字 like，如下所示：

```
mysql> select author from items
    -> where author like 'Dav%';
+-----------------------------------------------------+
| author                                              |
+-----------------------------------------------------+
| David J. Emberton, J. Scott Hamlin, David Emberton  |
| David Buser, Chris Ullman, Jon Duckett?             |
+-----------------------------------------------------+
2 rows in set (0.00 sec)
```

其中'Dav%'表示所有以'Dav'开头的字符串都会被检索，而如果想查询以'pig'结尾的字符串，可以使用'%pig'来查询，如果是想查询中间为'pig'的字符串，可以使用'%pig%'来查询。

（4）AND 关键字。AND 关键字主要用于连接多个 Where 查询子句，很多时候，查询需要多个条件成立才可以。例如要查询图书书名为包含“PHP”，同时图书编号小于“9”的书籍记录，可以使用 AND 关键字连接两个查询字符串。

```
mysql> select item_id,name from
    -> items where name like '%php%'
    -> and item_id<9;
+---------+--------------------------------+
| item_id | name                           |
+---------+--------------------------------+
|       5 | PHP and MySQL Web Development  |
|       6 | MySQL & PHP From Scratch       |
+---------+--------------------------------+
2 rows in set (0.00 sec)
```

以上通过 AND 关键字连接了两个条件查询语句，当然可以同时并列多个查询条件。AND 关键字表示所有连接的条件都必须符合，如果一条不符合，则不会被检索到。

（5）OR 关键字。OR 关键字和 AND 关键字一样，也是连接多个查询条件。通过 OR 关键字连接的多个条件，只要其中的一个条件符合，就会被检索出来。

例如要检索图书编号等于 15 的书籍记录，或者书名为“php”且图书编号不为 9 的书籍记录，可以使用 OR 关键字，如下所示：

```
mysql> select item_id,name from
    -> items where (name like '%php%'
    -> and item_id<>9) or
    -> item_id=15;
+---------+--------------------------------+
| item_id | name                           |
+---------+--------------------------------+
|       5 | PHP and MySQL Web Development  |
|       6 | MySQL & PHP From Scratch       |
|      15 | Mastering ColdFusion 4.5       |
+---------+--------------------------------+
3 rows in set (0.01 sec)
```

以上同时使用了 AND 关键字和 OR 关键字，其中括号中的两个条件必须同时成立才可以生效，即书名中含有“php”且图书编号不能为 9。另外图书编号为 15 的记录，则也符合查询要求。

（6）使用字段别名关键字 AS。在进行数据查询的时候，为了方便显示或者便于用户理解，可以将查询的字段名称显示成想要的名称。例如在查询图书名称的时候，列名仅为“name”，一般人不会知道到底是什么名称，可以使用字段别名关键字 AS 来显示指定，如下所示：

```
mysql> select category_id,name as
    -> category_name from categories;
+-------------+--------------------+
| category_id | category_name      |
+-------------+--------------------+
|           1 | Programming        |
|           2 | Databases          |
|           3 | HTML & Web design  |
+-------------+--------------------+
3 rows in set (0.02 sec)
```

以上使用了“name as category_name”语句，则在显示结果的时候，会用 category_name 代替 name。尤其是进行多表查询的时候，如果两个表都有相同的字段，更需要使用别名来处理。

（7）从多个表中获取记录。前面介绍的都是简单的从一个表中获取记录，但是很多时候，尤其是按照规范的方法设计的数据库，在进行查询的时候，需要涉及到多表查询。

要执行多表操作，必须通过 SQL 语句将两个表关联起来，进行关联查询。例如现在要查询一些图书信息并且需要显示图书的分类名称，就需要进行多表查询。因为在图书信息表 items 中没有图书分类名称的信息，只有图书分类编号，因此要获得图书分类名称，必须同时在图书分类信息表（categories 表）中关联查询，如下所示：

```
mysql> select categories.name,item_id
    -> ,items.name from categories,items
    -> where categories.category_id=
    -> items.category_id and items.item_id=5;
+-------------+---------+--------------------------------+
| name        | item_id | name                           |
+-------------+---------+--------------------------------+
| Programming |       5 | PHP and MySQL Web Development  |
+-------------+---------+--------------------------------+
1 row in set (0.00 sec)
```

以上查询了书籍的图书分类名称，图书编号和图书名称。可以看到，在进行多表查询的时候，使用了“表名.字段”的格式，这样做是为了防止 MySQL 查询错误，因为很可能存在多个表的字段名称相同，如 items 表的书名字段为 name，而 categories 表的分类名称也为 name，为了区分，必须要在字段前面添加表的名称。如果字段名称在表中是惟一的，则可以省略表名。

在查询条件中，增加了一条“categories.category_id= items.category_id”，这个条件是关联两个表的，表示查询的条件是 items 表中的图书分类编号 category_id 必须和 categories 表中的图书分类编号一样。如果不这样关联，则出现的结果肯定不是我们想像的那样，如下所示：

```
mysql> select categories.name,item_id,
    -> items.name from categories,items
    -> where items.item_id=15;
+-------------------+---------+--------------------------+
| name              | item_id | name                     |
+-------------------+---------+--------------------------+
| Programming       |      15 | Mastering ColdFusion 4.5 |
| Databases         |      15 | Mastering ColdFusion 4.5 |
| HTML & Web design |      15 | Mastering ColdFusion 4.5 |
+-------------------+---------+--------------------------+
3 rows in set (0.11 sec)
```

从上面可以看出，如果不设定关联条件，则会出现意想不到的结果，此时会将所有的图书分类名称显示出来。

（8）多表查询。多表查询同两个表查询原理基本类似，只是在进行表关联的时候需要设置多个关联条件。为了演示多表查询，首先需要向 orders 表中插入一条记录（因为目前 bookstore.sql 文件中的 orders 表中没有记录），如下所示：

```
mysql> insert into orders (member_id,item_id,quantity)
    -> values (1,5,1);
Query OK, 1 row affected (0.05 sec)
```

以上向 orders 表插入了一条订购记录，接下来需要查询该订单中的会员名称、书籍名称以及书籍分类名称的信息，需要从多个表中进行查询，如下所示：

```
mysql> select orders.order_id,members.member_login,
    -> items.name as book_name,categories.name as
    -> category_name from orders,members,items,categories
    -> where orders.order_id=2 and orders.member_id=
    -> members.member_id and orders.item_id=items.item_id
    -> and items.category_id=categories.category_id;
+----------+--------------+-------------------------------+---------------+
| order_id | member_login | book_name                     | category_name |
+----------+--------------+-------------------------------+---------------+
|        2 | admin        | PHP and MySQL Web Development | Programming   |
+----------+--------------+-------------------------------+---------------+
1 row in set (0.00 sec)
```

以上的查询语句比较复杂，需要从 4 个表中来联合查询记录，同两个表的查询原理是一致的。

（9）查找不匹配记录。查找不匹配记录，在 SQL 中被称为左关联。在前面的介绍的查询中，都是与条件相匹配的记录被查询出来，但是很多时候需要查询没有匹配的记录。例如从来没有下订单的会员、从来没有卖出去的书籍等。

左关联就是在两个表之间指定的关联条件下匹配的数据行。如果右边的表中没有匹配行，结果中就会增加一行，该行右边的列为 NULL。左关联通过关键字“LEFT JOIN…ON”来实现，如下所示：

```
mysql> select members.member_id,members.member_login,
    -> orders.order_id from members left join orders
    -> on members.member_id=orders.member_id;
+-----------+--------------+----------+
| member_id | member_login | order_id |
+-----------+--------------+----------+
|         1 | admin        |        1 |
|         1 | admin        |        2 |
|         2 | guest        |     NULL |
+-----------+--------------+----------+
3 rows in set (0.00 sec)
```

以上通过左关联查询所有客户的订单记录，从结果中可以看出，虽然会员“guest”从来没有订购记录，但是还是被列出来了。

如果想查询从来没有订购记录的客户，则只要增加一个判断语句即可，如下所示：

```
mysql> select members.member_id,members.member_login,
    -> orders.order_id from members left join orders
    -> on members.member_id=orders.member_id where
    -> orders.order_id is null;
+-----------+--------------+----------+
| member_id | member_login | order_id |
+-----------+--------------+----------+
|         2 | guest        |     NULL |
+-----------+--------------+----------+
1 row in set (0.02 sec)
```

从上面可以看出，只需要增加条件判断语句“where orders.order_id is null”即可查询出所有下过订单的会员了。

以上“members.member_id=orders.member_id”条件从句，还可以使用另外一种表达方法，即用 using 关键字，如下所示：

```
mysql> select members.member_id,members.member_login,
    -> orders.order_id from members left join orders
    -> using (member_id) where
    -> orders.order_id is null;
+-----------+--------------+----------+
| member_id | member_login | order_id |
+-----------+--------------+----------+
|         2 | guest        |     NULL |
+-----------+--------------+----------+
1 row in set (0.00 sec)
```

如果被查询的两个表的关键字段，即“member_id”是一样的，则可以省略“members.member_id=orders.member_id”语句，用“using (member_id)”代替。

（10）通过 ORDER BY 对记录进行排序。数据库中的数据可能是杂乱无章的，很多时候，需要查询显示的结果按照一定的字段或者原则进行排序，此时可以使用 ORDER BY 进行记录排序。

如果要对 items 表中的图书价格字段 price 进行排序，可以进行如下操作：

```
mysql> select item_id,price from items
    -> where item_id<5 order by price;
+---------+-------+
| item_id | price |
+---------+-------+
|       3 | 15.19 |
|       2 | 39.96 |
|       1 | 39.99 |
|       4 | 39.99 |
+---------+-------+
4 rows in set (0.03 sec)
```

以上按照升序对图书的价格进行排序，如果要对图书价格进行降序排列，则可以用关键

字 DESC，如下所示：

```
mysql> select item_id,price from items
    -> where item_id<5 order by price DESC;
+---------+-------+
| item_id | price |
+---------+-------+
|       1 | 39.99 |
|       4 | 39.99 |
|       2 | 39.96 |
|       3 | 15.19 |
+---------+-------+
4 rows in set (0.00 sec)
```

另外可以同时对多个字段进行排序，如下所示：

```
mysql> select item_id,price from items
    -> where item_id<5 order by price,item_id DESC;
+---------+-------+
| item_id | price |
+---------+-------+
|       3 | 15.19 |
|       2 | 39.96 |
|       4 | 39.99 |
|       1 | 39.99 |
+---------+-------+
4 rows in set (0.00 sec)
```

从结果可以看出，如果不对 item_id 进行降序排列，默认的情况下，出现的结果是 3、2、1、4（item_id 字段），而如果增加对 item_id 排序，则结果不同。多字段排序，是按照先后顺序进行的，在本例中，首先是按照 price 进行排序，然后再对 item_id 进行排序。

（11）常用的统计函数。在操作数据库的时候，经常需要获取数据库记录中的一些统计数据，比如平均值、最大值、最小值和记录数等。

常用的统计函数如表 8-14 所示。

表 8-14　常用统计函数

名　称	说　明
AVG（字段名）	获取指定列的平均值
COUNT（字段名）	如果指定一个字段，则会统计该字段中非空记录的数量。如果在前面增加 DISTINCT，则会统计不同值的记录数，相同值的当作一条记录。如果使用 COUNT(*)则会统计包含空值的所有记录数。
MIN（字段名）	获取指定字段的最小值
MAX（字段名）	获取指定字段的最大值
STD（字段名）	指定字段的标准背离值
STDDEV（字段名）	与 STD 相同
SUM（字段名）	指定字段所有记录的总和

要统计 items 表中 item_id 的最大值和最小值以及价格的最大值和最小值，如下所示：

```
mysql> select min(item_id) min_id,max(item_id) as max_id,
    -> min(price) as min_price,max(price) as max_price
    -> from items;
+--------+--------+-----------------+-----------------+
| min_id | max_id | min_price       | max_price       |
+--------+--------+-----------------+-----------------+
|      1 |     22 | 15.189999580383 | 47.990001678467 |
+--------+--------+-----------------+-----------------+
1 row in set (0.00 sec)
```

接下来统计平均价格、价格总数以及总记录数，如下所示：

```
mysql> select avg(price) as avg_price,
    -> sum(price) as sum_price,
    -> count(*) as total_record from items;
+-----------------+-----------------+--------------+
| avg_price       | sum_price       | total_record |
+-----------------+-----------------+--------------+
| 33.264091274955 | 731.81000804901 |           22 |
+-----------------+-----------------+--------------+
1 row in set (0.02 sec)
```

（12）通过 GROUP BY 分组显示记录。很多时候，需要对有相同性质的记录进行分组统计查询，如统计每个作者写作的书籍数量、每个会员采购记录等，此时通过分组查询比较方便，分组查询是通过 GROUP BY 关键字进行的。

如果要按照价格来分类统计每种价格的书籍种类，可以通过如下方法：

```
mysql> select count(*),price from items
    -> group by price;
+----------+-------+
| count(*) | price |
+----------+-------+
|        1 | 15.19 |
|        1 | 15.99 |
|        1 | 19.99 |
|        1 | 23.96 |
|        1 | 23.99 |
|        2 | 27.96 |
|        2 | 31.99 |
|        1 | 34.99 |
|        1 | 35.96 |
|        2 |    36 |
|        1 | 39.95 |
|        1 | 39.96 |
|        5 | 39.99 |
|        1 | 41.99 |
|        1 | 47.99 |
+----------+-------+
15 rows in set (0.02 sec)
```

以上按照价格进行统计，从中可以看出，价格为 39.99 元的书有 5 本，36 元的有 2 本……。

（13）HAVING 子句。前面介绍 GROUP BY 的用法，如果需要针对 GROUP BY 设置限制条件，则不能使用 Where 子句，必须使用 HAVING 子句来设置。

例如要按照价格超过 35 元的书籍进行统计，必须要设定条件，如下所示：

```
mysql> select count(*),price from items
    -> group by price having price>35;
+----------+-------+
| count(*) | price |
+----------+-------+
|        1 | 35.96 |
|        2 |    36 |
|        1 | 39.95 |
|        1 | 39.96 |
|        5 | 39.99 |
|        1 | 41.99 |
|        1 | 47.99 |
+----------+-------+
7 rows in set (0.00 sec)
```

（14）LIMIT 关键字。通过 LIMIT 关键字可以设定要返回的记录数。该关键字不是标准 SQL 中的，而是 MySQL 的扩展标准，因此采用该关键字不能适用其他数据库。

LIMIT 的用法如下所示：

```
mysql> select item_id,price from items
    -> limit 2,3;
+---------+-------+
| item_id | price |
+---------+-------+
|       3 | 15.19 |
|       4 | 39.99 |
|       5 | 39.99 |
+---------+-------+
3 rows in set (0.00 sec)
```

以上表示从查询的记录集中的第 2 行开始（注意行的索引是从 0 开始的，实际上是第 3 条记录）显示 3 条记录。

（15）使用子查询。所谓子查询就是嵌套在另外一个查询内部的查询。尽管大多数的子查询都可以通过连接和临时表来实现，但是通过子查询则更容易理解和编写。

子查询最常见的用法就是将一个子查询的结果作为另外一个查询的条件，如下所示：

```
mysql> select item_id,name from items
    -> where item_id=(select max(item_id)
    -> from items);
+---------+----------------------------------------+
| item_id | name                                   |
+---------+----------------------------------------+
|      22 | C# - Programming with the Public Beta |
+---------+----------------------------------------+
1 row in set (0.01 sec)
```

以上查询出图书编号 item_id 最大的图书的信息。

在进行子查询的时候，需要用到操作符，如 ANY、IN、SOME 或者 ALL。具体含义如表 8-15 所示。

表 8-15　常用操作符

名　称	示　例	说　明
ANY	SELECT field1 FROM table1 where field1 > ANY (SELECT field1 FrOM table2)	如果同子查询的任何记录比较都为真，则返回 TRUE
IN	SELECT field1 FROM table1 where field1 IN (SELECT field1 FrOM table2)	等价于 =ANY
SOME	SELECT field1 FROM table1 where field1 > SOME (SELECT field1 FrOM table2)	和 ANY 含义一样
ALL	SELECT field1 FROM table1 where field1 > ALL (SELECT field1 FrOM table2)	如果同子查询的全部记录比较为真，则返回 TRUE

下面演示一个 IN 操作符的应用，如下所示：

```
mysql> select member_id,member_login from members
    -> where member_id in (select distinct
    -> member_id from orders);
+-----------+--------------+
| member_id | member_login |
+-----------+--------------+
|         1 | admin        |
+-----------+--------------+
1 row in set (0.03 sec)
```

以上查询所有下过订单的会员的信息。子查询是从订单表中查询有订单的会员编号 member_id，关键字 DISTINCT 表示去掉重复的记录，因为有的会员可能多次下过订单。

（16）关联子查询。所谓关联子查询中，指可以在内部查询中使用外部查询的结果，如下所示：

```
mysql> select items.item_id,name from items
    -> where exists (select * from orders
    -> where orders.item_id=items.item_id);
+---------+-----------------------------------------+
| item_id | name                                    |
+---------+-----------------------------------------+
|       1 | Web Database Development : Step by Step |
|       5 | PHP and MySQL Web Development           |
+---------+-----------------------------------------+
2 rows in set (0.03 sec)
```

以上查询了所有有被订购记录的书籍信息。这里使用了关键字 EXISTS，如果查询还没有订购记录的书籍信息，可以使用 NOT EXIST（细心的读者会发现通过左关联也可以实现）。

在上面的查询中，子查询中使用了外部查询的表集 items，这就是关联子查询。如果外部查询中没有包含的表，则不能引用。

（17）将子查询作为临时表。有些时候，可以将子查询的结果作为一个临时的表放在 FROM 后面来进行查询，如下所示：

```
mysql> select member_id from (select * from members
    -> where member_login='admin') as tempTable;
+-----------+
| member_id |
+-----------+
|         1 |
+-----------+
1 row in set (0.03 sec)
```

以上是将“select * from members where member_login='admin'”子查询作为一个临时表，注意在使用临时表的时候，必须要为该子查询创建一个别名。

SELECT 也可以被用于检索没有引用任何表的计算列。如下所示：

```
mysql> select 10+10;
+-------+
| 10+10 |
+-------+
|    20 |
+-------+
1 row in set (0.00 sec)
```

8.2.3 编辑记录

前面介绍了数据的插入和查询，接下来介绍如何在数据库中编辑数据。通过 UPDATE 语句可以实现，该语句的格式如下：

```
UPDATE [LOW_PRIORITY] [IGNORE] tbl_name
   SET col_name1=expr1 [, col_name2=expr2 ...]
   [WHERE where_definition]
   [ORDER BY ...]
   [LIMIT rows]
```

或者：

```
UPDATE [LOW_PRIORITY] [IGNORE] tbl_name [, tbl_name ...]
   SET col_name1=expr1 [, col_name2=expr2 ...]
   [WHERE where_definition]
```

UPDATE 以新的值更新现存表中行的列。SET 子句指出要修改哪个列和它们应该给定的值。WHERE 子句如果被给出，指定哪个记录行应该被更新。否则，所有的记录行被更新。如果 ORDER BY 子句被指定，记录行将被以指定的次序更新。

如要将订单表 orders 中的订单数量全部改成 5，如下所示：

```
mysql> update orders set quantity=5;
Query OK, 2 rows affected (0.11 sec)
Rows matched: 2  Changed: 2  Warnings: 0
```

以上显示两条记录被修改了，如果此时查看 orders 表，会发现订单表中所有的订单数量都被改成 5 了，如下所示：

```
mysql> select * from orders;
+----------+-----------+---------+----------+
| order_id | member_id | item_id | quantity |
+----------+-----------+---------+----------+
|        1 |         1 |       1 |        5 |
|        2 |         1 |       5 |        5 |
+----------+-----------+---------+----------+
2 rows in set (0.00 sec)
```

以上通过 UPDATE 修改记录的时候，没有设定条件，所以将所有的记录都修改了，如果指定了限制条件，则只会修改符合条件的记录，如下所示：

```
mysql> update orders set quantity=1
    -> where order_id=1;
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

以上将 order_id 等于 1 的订购数量 quantity 修改成 1，而 order_id 等于 2 的记录不会被改变，从其返回的修改记录数量“1 row affected”也可以看出，结果如下：

```
mysql> select * from orders;
+----------+-----------+---------+----------+
| order_id | member_id | item_id | quantity |
+----------+-----------+---------+----------+
|        1 |         1 |       1 |        1 |
|        2 |         1 |       5 |        5 |
+----------+-----------+---------+----------+
2 rows in set (0.00 sec)
```

如果指定关键词 LOW_PRIORITY，UPDATE 的执行将被延迟，直到没有其他的客户端正在读取表。

如果指定关键词 IGNORE，该更新语句将不会异常终止，即使在更新过程中出现重复键错误。导致冲突的记录行将不会被更新。

如果在一个表达式中从 tbl_name 中访问一个列，UPDATE 使用列的当前值。比如想要将订单数量全部增加 1，可以使用如下语句：

```
mysql> update orders set quantity=quantity+1;
Query OK, 2 rows affected (0.05 sec)
Rows matched: 2  Changed: 2  Warnings: 0
```

以上就直接引用了字段 quantity 当前的值，然后在当前值上面增加 1，查询结果如下：

```
mysql> select * from orders;
+----------+-----------+---------+----------+
| order_id | member_id | item_id | quantity |
+----------+-----------+---------+----------+
|        1 |         1 |       1 |        2 |
|        2 |         1 |       5 |        6 |
+----------+-----------+---------+----------+
2 rows in set (0.00 sec)
```

从上面可以看出，quantity 的数量都增加了 1。

UPDATE 赋值是从左到右计算的。首先看下面的语句：

```
mysql> update orders set quantity=quantity*2,
    -> quantity=quantity+1;
```

```
Query OK, 2 rows affected (0.03 sec)
Rows matched: 2  Changed: 2  Warnings: 0
```

以上 UPDATE 语句同时进行了两次编辑，首先将当前记录乘以 2，然后在结果上再增加 1。在执行的时候，是按照从左到右的顺序进行的，如下所示：

```
mysql> select * from orders;
+----------+-----------+---------+----------+
| order_id | member_id | item_id | quantity |
+----------+-----------+---------+----------+
|        1 |         1 |       1 |        5 |
|        2 |         1 |       5 |       13 |
+----------+-----------+---------+----------+
2 rows in set (0.00 sec)
```

从上面的结果可以看出，quantity 首先被乘 2，然后再增加了 1。

UPDATE 返回实际被改变的记录行数目。函数 mysql_info()返回被匹配并更新的记录行数目，以及在 UPDATE 期间发生的警告的数目。

可以使用 LIMIT #来确保只有给定的记录行数目被更改。

```
mysql> update orders set quantity=1 limit 1;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

以上语句将第一条记录的 quantity 修改成了 1，如下所示：

```
mysql> select * from orders;
+----------+-----------+---------+----------+
| order_id | member_id | item_id | quantity |
+----------+-----------+---------+----------+
|        1 |         1 |       1 |        1 |
|        2 |         1 |       5 |       13 |
+----------+-----------+---------+----------+
2 rows in set (0.00 sec)
```

如果一个 ORDER BY 子句被使用，记录行将以指定的次序被更新。只有连同 LIMIT 一起才有用。

同时用户也可以执行一个包含多个表的 UPDATE 的操作：

```
UPDATE table_1,table_2 SET table_1.field_1=table_2.field_1 WHERE
table_1.id=table_2.id;
```



注意 多表 UPDATE 不可以使用 ORDER BY 或 LIMIT。

8.2.4 删除记录

当出现错误数据或者某些数据已经失去意义的时候，需要将数据从数据库中删除。删除数据使用 DELETE 语句来实现，该语句的格式如下：

```
DELETE [LOW_PRIORITY] [QUICK] FROM table_name
      [WHERE where_definition]
      [ORDER BY ...]
      [LIMIT rows]
```

或者

```
DELETE [LOW_PRIORITY] [QUICK] table_name[.*] [, table_name[.*] ...]
      FROM table-references
      [WHERE where_definition]
```

或者：

```
DELETE [LOW_PRIORITY] [QUICK]
```

```
        FROM table_name[.*] [, table_name[.*] ...]
        USING table-references
        [WHERE where_definition]
```

DELETE 从 table_name 中删除 where_definition 中给定条件的记录行，并返回删除的记录数目。

如要删除 orders 表中的第 1 条记录，则可以使用下面的语句：

```
mysql> delete from orders where order_id=1;
Query OK, 1 row affected (0.03 sec)
```

从执行结果可以看出，1 条记录被删除了。

如果发出一个没有 WHERE 子句的 DELETE，所有的记录行将被删除。如果以 AUTOCOMMIT 模式执行它，那么类似于 TRUNCATE。

下面的语句将删除 orders 表中的所有记录，由于没有使用 WHERE 条件，所以不知道到底被删除了多少条记录，结果返回 0，如下所示：

```
mysql> delete from orders;
Query OK, 1 row affected (0.01 sec)
```

此时如果查询该表，则会显示没有记录。

当然如果使用 DELETE 删除整个表的数据效率并不高，推荐使用 TRUNCATE 语句，该方法可以快速地删除表中的内容。

因为前面已经删除了 orders 表中的记录，为了方便演示，下面插入两条记录，如下所示：

```
mysql> insert into orders (member_id,item_id,quantity)
    -> values (1,1,1);
Query OK, 1 row affected (0.06 sec)

mysql> insert into orders (member_id,item_id,quantity)
    -> values (2,2,2);
Query OK, 1 row affected (0.03 sec)
```

接着进行查询，如下所示：

```
mysql> select * from orders;
+----------+-----------+---------+----------+
| order_id | member_id | item_id | quantity |
+----------+-----------+---------+----------+
|        1 |         1 |       1 |        1 |
|        2 |         2 |       2 |        2 |
+----------+-----------+---------+----------+
2 rows in set (0.00 sec)
```

可以看出，两条记录已经被插入，接下来使用 TRUNCATE 语句删除数据库表数据，如下所示：

```
mysql> truncate orders;
Query OK, 2 rows affected (0.05 sec)
```

如果删除时指定关键词 LOW_PRIORITY，DELETE 的执行将被延迟，直到没有其他的客户端正在读取表。

如果指定关键词 QUICK，那么在删除过程中存储引擎将不会归并索引页，这可能会加速某些类型的删除操作。

在 MySQL 中，删除的记录被放在一个链接表中维护，以后的 INSERT 操作将重新使用删除后的记录位置。为了回收闲置的空间，并减小文件尺寸，使用 OPTIMIZETABLE 语句或 myisamchk 实用程序重新组织表。OPTIMIZE TABLE 使用比较容易，但是 myisamchk 更快点。

如果一个 ORDER BY 子句被使用，记录行将以指定的次序删除，只有连同 LIMIT 一起才有用。

DELETE 语句的 LIMIT rows 选项是 MySQL 特有的，它告诉服务器在控制权被返回到客户端之前可被删除的最大记录行数目。这可以用来确保一个特定的 DELETE 命令不会占用太长的时间。可以简单地重复使用 DELETE 命令，直到被影响的记录行数目小于 LIMIT 值。

从 MySQL 4.0 开始，在 DELETE 语句中可以指定多个表，用于从一个表中删除依赖于多表中的特殊情况的记录行。然而，在多表删除中，不能使用 ORDER BY 或 LIMIT。

8.3 MySQL 的高级应用

在接下来介绍 MySQL 数据库的高级应用，包括存储过程的创建和应用等内容。

8.3.1 事务

事务是数据库高级编程的一个重要内容。关于事务的概念以及重要性，读者可以参考相关的书。本书只是介绍如何在 MySQL 中实现事务处理。

在默认的情况下，MySQL 对于普通的 SQL 语句都是自动提交的。 也就是当 SQL 语句执行完毕后就会立刻在数据库中得到提交。

在 MySQL 中，如果要关闭自动提交，可以使用以下语句：

```
set autocommit=0;
```

当自动提交被关闭之后，执行 SQL 语句则会自动启动事务处理，所有的 SQL 语句都会被当作事务来执行。

如果自动提交打开的时候，系统是不会自动进入事务处理阶段的，必须使用如下语句才能启动事务处理：

```
start transaction;
```

在执行完事务块的所有语句后，如果要真正提交到数据库，必须调用以下语句：

```
commit;
```

如果在事务的执行过程中出现意外或者发生错误，可以取消所有执行的 SQL 语句，调用以下语句即可：

```
rollback;
```

当真正调用 commit 语句后，执行结果才能够被其他用户看见。

下面通过一个实例来学习和验证事务的作用和处理流程，代码如下：

```
--启动事务
start transaction;
--向 oders 表中插入一条记录
insert into orders (member_id,item_id,quantity) values (1,1,1);

insert into orders (member_id,item_id,quantity)  values (2,2,2);

--此时查询数据库表
select * from orders;  --此时会查询到结果，但是实际上该结果并没有被插入到数据库
```

为了方便演示重新启动一个 MySQL 客户端，输入如下内容：

```
select * from orders;
```

此时看不到记录，这说明事务已经启动，但是还没有提交。

如果在第一个客户端执行“commit;”命令，则可以在其他客户端查看到被插入的记录。

启动事务以及插入数据过程如图 8-8 所示。

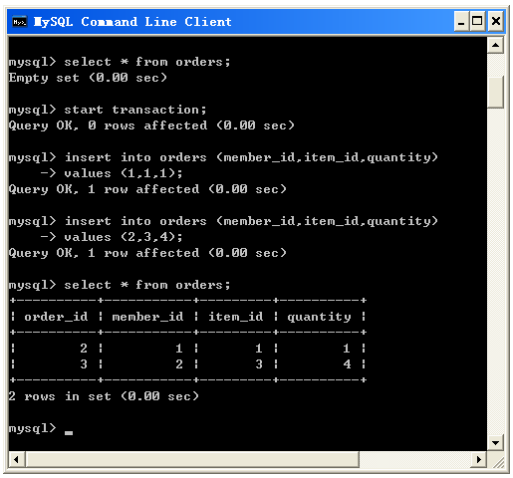


图 8-8　启动事务并插入记录

从图 8-8 中可以看出，对同一个客户来说，虽然事务没有提交，但是还是可以从数据库表 orders 中检索记录，但是实际上并没有提交到数据库中，只是在缓存中。

另外启动一个客户端界面，执行查询语句，如图 8-9 所示。

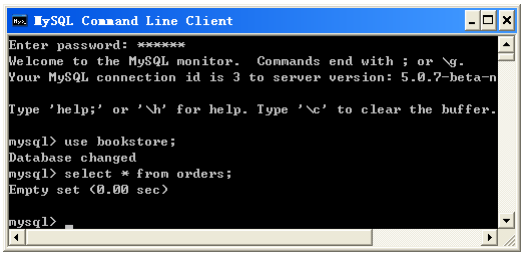


图 8-9　另一个客户端

从图 8-9 中可以看出，虽然在图 8-8 中插入了两条记录，但是因为事务没有提交，所以其他客户并看不到。接下来在图 8-8 中执行事务提交，如图 8-10 所示。

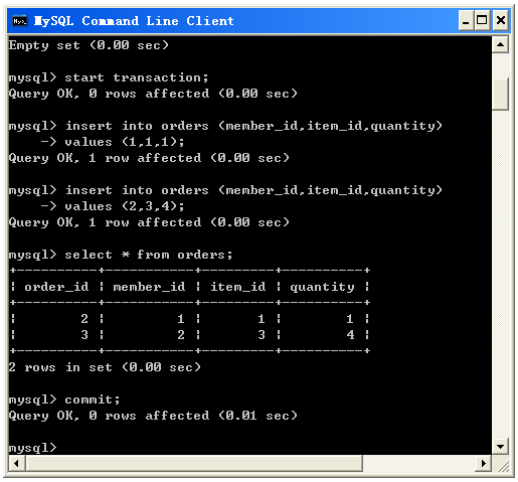


图 8-10　事务提交

此时在第二个客户端查询 orders 表，会发现记录已经被提交了，如图 8-11 所示。

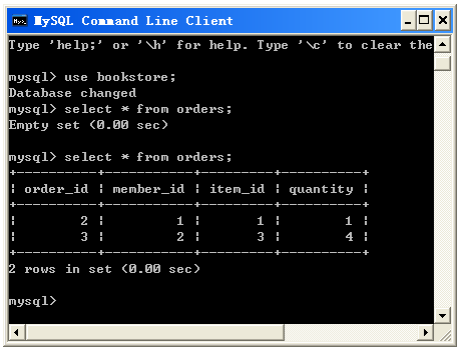


图 8-11　事务提交后，第二个客户端的查询结果



如果不启动事务，则在第一个客户端插入记录后，其他客户是可以立刻看到数据库变化的，如图 8-12 和 8-13 所示。

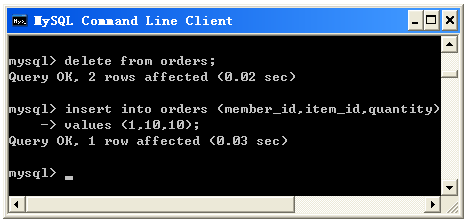


图 8-12　第一个客户端插入记录

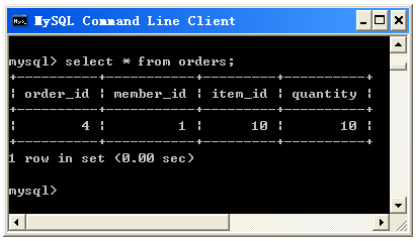


图 8-13　第二个客户端立刻看到结果

如果在事务处理过程中出现错误，可以调用“rolback;”语句，取消所有的操作，具体应用过程和“commit”语句类似。

8.3.2　存储过程

存储过程是数据库编程中最重要的内容。虽然以前的 MySQL 并不支持存储过程，但是 MySQL 5.0 已经开始支持存储过程。这样可以大大提高数据库的处理速度，同时也可以提高数据库编程的灵活性。

存储过程通常有以下优点。

（1）存储过程允许标准组件式编程。存储过程被创建以后，可以在程序中被多次调用，而不必重新编写该存储过程的 SQL 语句，而且数据库专业人员可随时对存储过程进行修改，而对应用程序源代码毫无影响。

（2）存储过程能够实现较快的执行速度。如果某一操作包含大量的 Transaction-SQL 代码或分别被多次执行，那么存储过程要比批处理的执行速度快很多。因为存储过程是预编译的，

在首次运行一个存储过程时查询，优化器对其进行分析优化，并给出最终被存在系统表中的执行计划。而批处理的 Transaction-SQL 语句在每次运行时都要进行编译和优化，速度相对要慢一些。

（3）存储过程能够减少网络流量。针对同一个数据库对象的操作（如查询、修改），如果这一操作所涉及到的 Transaction-SQL 语句被组织成存储过程，那么当在客户计算机上调用该存储过程时，网络中传送的只是该调用语句，从而大大增加了网络流量降低网络负载。

（4）存储过程可被作为一种安全机制来充分利用。系统管理员通过对执行某一存储过程的权限进行限制，能够实现对相应的数据访问权限的限制，避免非授权用户对数据的访问，保证数据的安全。

下面介绍如何在 MySQL 中实现存储过程。

1．创建存储过程

声明一个存储过程的语法如下：

```
CREATE PROCEDURE sp_name ([proc_parameter[,...]])
   [characteristic ...] routine_body
```

其中 CREATE PROCEDURE 是创建存储过程的关键字，sp_name 表示要创建存储过程的名称，proc_parameter 表示存储过程的参数，该参数可以是 MySQL 合法的数据类型，而 routine_body 是任何合法 SQL 语句。

下面举一个简单的例子来说明，如下所示：

```
mysql> delimiter //
mysql> CREATE PROCEDURE simpleproc (OUT param1 INT)
    -> BEGIN
    ->   SELECT COUNT(*) INTO param1 FROM items;
    -> END
    -> //
mysql> delimiter ;
mysql> CALL simpleproc(@a);
mysql> select @a;
```

下面依次介绍以上语句的含义。

“delimiter //”表示将当前的语句结束符号从“;”改成“//”，这样就可以在存储过程内部使用分号“;”了。

“CREATE PROCEDURE simpleproc（OUT param1 INT）”表示创建一个存储过程，名称为“simpleproc”，而参数“OUT param1 INT”表示该存储过程有一个参数，参数的类型是整型 INT，OUT 表示该参数是一个输出参数。

提示　在存储过程中，参数类型有 3 种，除了上面的 OUT 之外，还可以使用 IN，表示该参数必须要传入值存储过程，另外还有一种类型就是 INOUT，表示该参数既可以为输入参数也可以为输出参数。使用时应该根据需要来设定。

接下来是 BEGIN 关键字，它和 END 是配合使用的，所有的存储过程的 SQL 语句都必须包含在 BEGIN…END 中。

该存储过程比较简单，只有一个 SQL 语句，该语句表示将统计的记录数保存到参数 param1 中，将数据保存到变量中使用 INTO 语句。

“//”表示要创建该存储过程，因为第一行已经将执行 SQL 语句的结束符改成了“//”。

创建完成以后，接着需要恢复分隔符为分号“;”，所以使用“delimiter ;”。

语句“CALL simpleproc(@a);”表示调用该存储过程，@a 是传入的参数，用来保存统计结果。

最后一句是将存储过程的结果显示出来。

整个存储过程的执行如图 8-14 所示。

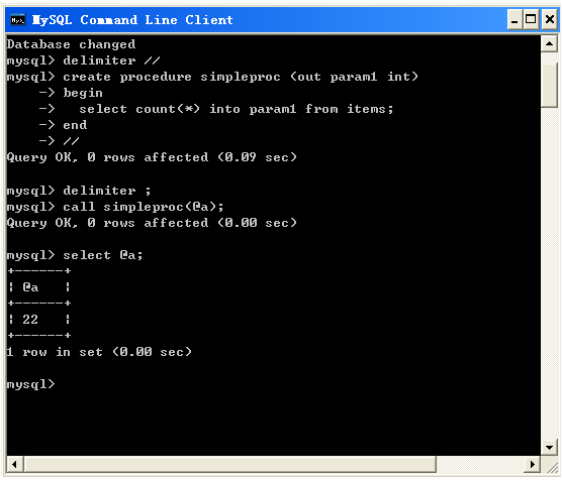


图 8-14　存储过程的执行过程

从图 8-14 中可以看出，通过“select @a;”就可以直接获得 items 表的记录数量。当然存储过程还可以做很多复杂的事情。

2．创建函数

在介绍完如何创建存储过程之后，有必要介绍一下如何在 MySQL 中使用函数。创建一个函数的语法同创建存储过程基本类似，如下所示：

```
CREATE FUNCTION sp_name ([func_parameter[,...]])
  RETURNS type
  [characteristic ...] routine_body
```

sp_name 为函数名称，func_parameter 为函数的参数，routine_body 为函数的语句内容。

为了方便读者理解，下面介绍一个简单实例。

```
mysql> delimiter //
mysql> CREATE FUNCTION hello(s CHAR(20)) RETURNS CHAR(50)
    -> RETURN CONCAT('Hello, ',s,'!');
    -> //
mysql> delimiter ;
mysql> SELECT hello('world');
```

在以上代码中，函数名称为 hello，而 s 表示参数，char(20)表示参数的类型，return char(50)表示返回值的类型是 char。

该函数只有一个语句，即返回一个字符串，该字符串是将“hello”和参数 s 合并成一个字符串。调用函数使用“SELECT hello('world');”语句。



创建函数和创建存储过程是有区别的，创建存储过程必须使用 BEGIN…END，而创建函数并不一定，如果只有一行语句则不需要，如果多行，则需要。函数的参数没有 IN 或者 OUT 类型，因为所有的函数参数都是 IN 类型的，因此创建的时候不需要声明。

整个函数的创建以及执行过程如图 8-15 所示。

3．局部变量

在编写很多复杂的存储过程以及复杂的函数的时候，需要临时变量。接下来将介绍如何在 MySQL 中声明并使用一个临时变量。

局部变量的声明格式如下所示：

```
DECLARE var_name[,...] type [DEFAULT value]
```

其中 var_name 表示变量名称，可以同时声明多个变量。type 表示变量类型，而 DEFAULT value 表示设置默认值，如 DEFAULT 0.01。局部变量声明必须在 BEGIN…END 之间。

如果要给一个变量赋值，格式如下：

```
SET var_name = expr [, var_name = expr] ...
```

赋值使用 SET 关键字，值可以是一个具体的值，也可以是一个表达式或者变量。

下面通过一个实例来学习。

```
mysql> delimiter //
mysql> create procedure sp_1(in id int)
    -> begin
    -> declare cname varchar(20);
    -> declare xid int;
    -> select name,category_id into cname,xid
    -> from categories where category_id=id;
    -> select cname,xid;
    -> end
    -> //
mysql> delimiter ;
mysql> call sp_1(1);
```

以上代码中，id 是一个传输参数。其中声明了两个局部变量，分别是 cname 和 xid，cname 用来保存从 categories 查询出来的 name 和 category_id。id 用做查询条件。其中“select cname，xid；”表示执行存储过程后，要显示的结果。

图 8-16 所示为整个存储过程的执行结果。

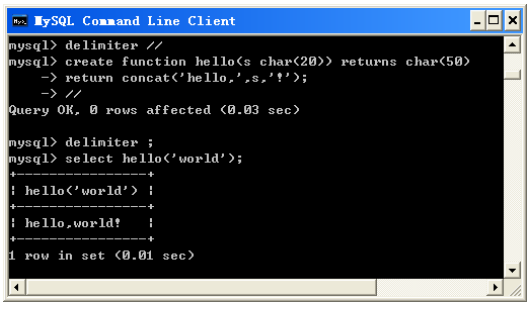


图 8-15　在 MySQL 中使用函数

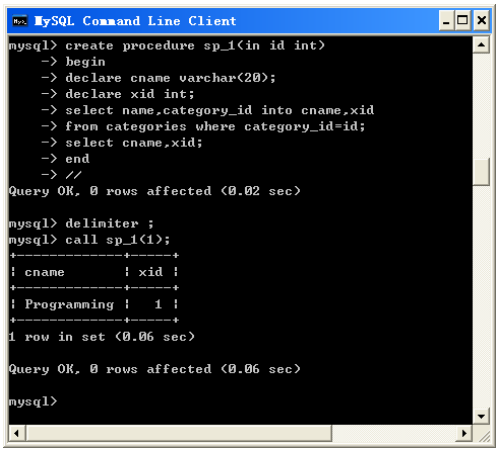


图 8-16　局部变量的应用

由于篇幅有限，读者如果需要掌握更多的存储过程知识，可以参考 MySQL 的帮助手册，学习诸如游标、SQL 语句的控制结构等内容。

8.4 使用 MySQL Administrator 管理数据库

可能有人不习惯 MySQL 数据库的命令行操作方式，更喜欢可视化操作界面。低版本的 MySQL 提供了一个可视化的小工具 winmysqladmin.exe，此工具在数据库安装路径的 bin 目录下。可能是开发者觉得用处不大，在高版本中这个工具就不提供了，读者可以在 4.0 版本中找到。

此外，可以使用 MySQL Administrator 工具来管理数据库，该工具对数据库建立和维护提供了友好的可视化操作界面。

MySQL Administrator 可以在 MySQL 的官方网站下载，地址是 http://dev.mysql.com/downloads/administrator/1.1.html。目前能够支持 MySQL5.0 的稳定版本是 1.1。下载界面如图 8-17 所示。

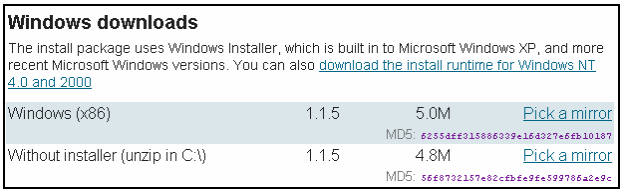


图 8-17 MySQL Administrator 下载界面

下载第一项，下载后的文件名为 mysql-administrator-1.1.5-win.msi。双击安装，如图 8-18 所示。

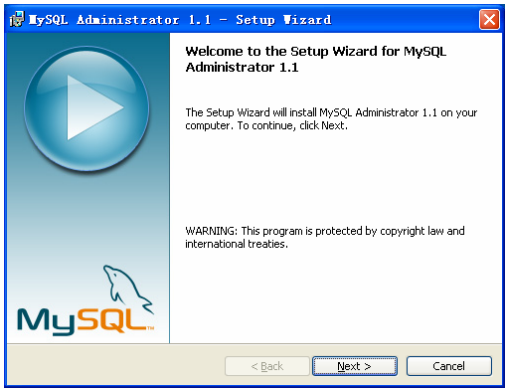


图 8-18 MySQL Administrator 安装界面

MySQL Administrato 的安装比较简单，依次单击“Next”按钮即可。最好修改默认安装路径，把它和 MySQL 数据库安装在同一目录下，这样方便管理。下面简单介绍该软件的应用。

（1）安装好以后，选择【开始→所有程序→MySQL→MySQL Administrater】，启动 Administrater，如图 8-19 所示。

（2）在“Server Host”一栏填上 localhost，默认端口为 3306，在“Username”一栏填上用户名“root”，在“Password”一栏填上密码“123456”（根据自己实际设置填写）。点击【OK】按钮，出现如图 8-20 所示界面。

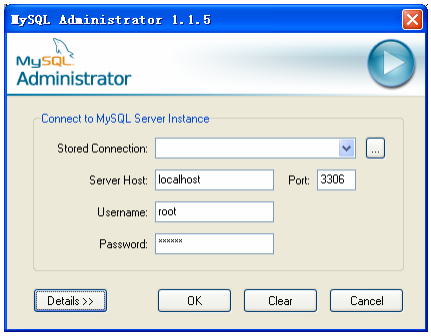


图 8-19　MySQL Administrator 登录界面

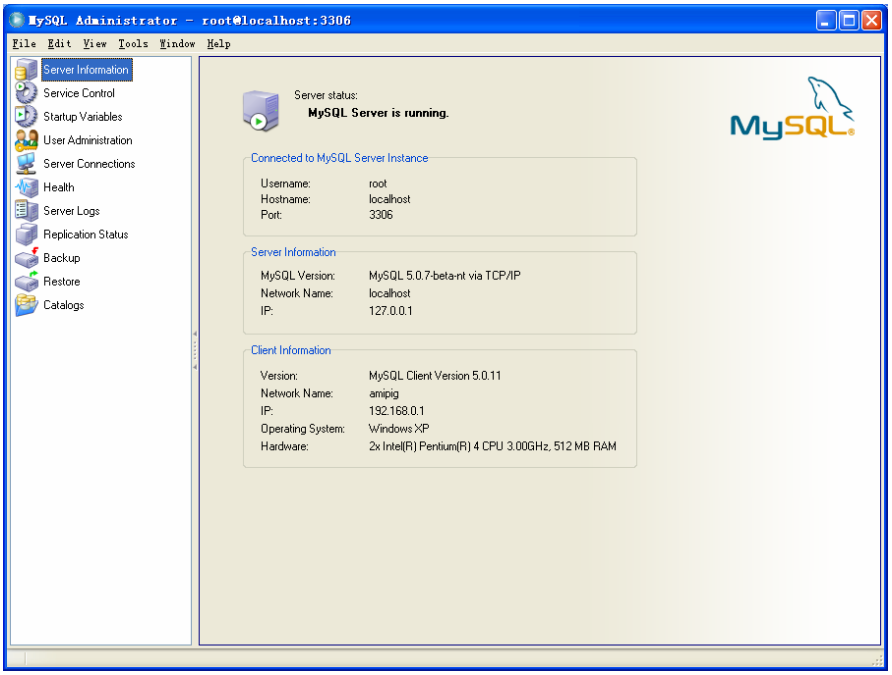


图 8-20　MySQL Administrator 管理界面

（3）在 MySQL 管理器中，能够图形化的进行各种管理任务。其中左侧是任务导航栏。依次包含以下功能模块。

Server Information：用于显示当前连接的 MySQL 服务器的一些常规信息，如操作系统、IP 地址、用户名和 MySQL 的版本等。

Service Control：用于控制服务器的开始和启动，通过该模块可以方便地启动和停止 MySQL 的服务，同时还可进行其他一些服务设置，如显示的名称、服务器设置文件路径等。

Startup Variables：用于设置 MySQL 的环境变量，如 TCP 端口号、缓存设置、日志设置和复制设置等。这些设置通常保存在 MySQL 的设置文件 my.ini 文件中。

User Administration：用于对当前 MySQL 用户的管理，并且可以针对不同的用户进行授权，相比较前面的命令行格式的授权，这里要容易很多。

Server Connection：用于查看当前所有的客户连接进程，管理员可以终止某些进程的执行。

Health：用于监控 MySQL 的运行情况。

Server Logs：用于 MySQL 的日志管理。

Replication Status：用于监控服务器的复制信息。

Backup：用于备份 MySQL 数据库。

Restore：用于还原 MySQL 数据库。

Catalogs：该模块是最常用的模块，用于对实际数据库的管理，如创建数据库、创建表、创建索引和创建存储过程等。

下面简要介绍 Catalogs 模块的使用。

用鼠标单击 Catalogs 模块，出现如图 8-21 所示的界面。

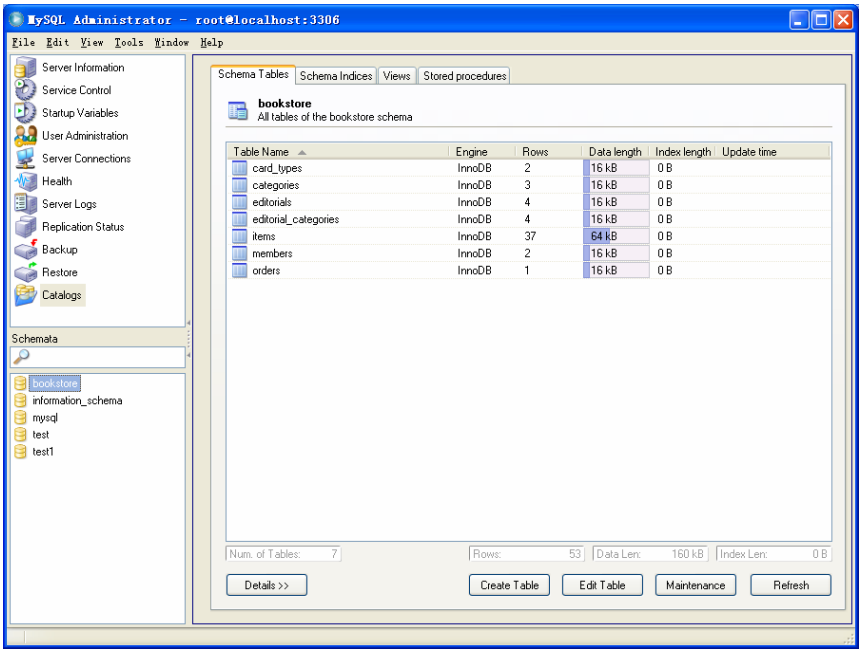


图 8-21　catalogs 模块

在图 8-21 中，左下侧显示的是当前系统所有创建好的数据库。其中 mysql 和 information_schema 数据库是系统自带的管理数据库。在前面章节中，已经创建了 bookstore 数据库，用鼠标单击，在右侧会显示该数据库所有已经创建的表。用户可以查看并编辑每个表的信息。

为了便于读者理解，下面重新利用该管理器创建一个新的数据库。

在图 8-21 左下侧，单击右键，弹出如图 8-22 所示的菜单。

单击菜单【Create New Schema】，弹出如图 8-23 所示的创建新数据库的界面。

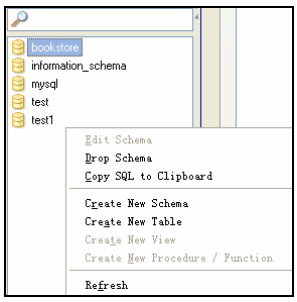


图 8-22　创建数据库的右键菜单

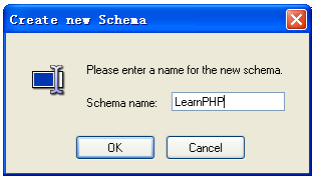


图 8-23　创建新数据库

在图 8-23 中，输入要创建的数据库名称，如“LearnPHP”，然后单击“OK”按钮，此时新创建的数据库就会出现在列表中，如图 8-24 所示。

选中新创建的 LearnPHP 数据库，在右侧的空白区域单击右键，弹出如图 8-25 所示的菜单。

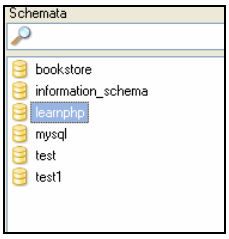


图 8-24　新建 LearnPHP 数据库

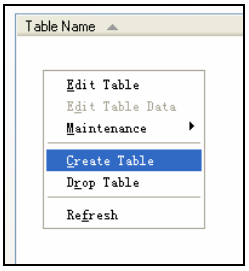


图 8-25　新建数据表菜单

在图 8-25 中，选中【Create Table】，将弹出如图 8-26 所示的数据库表设计界面。

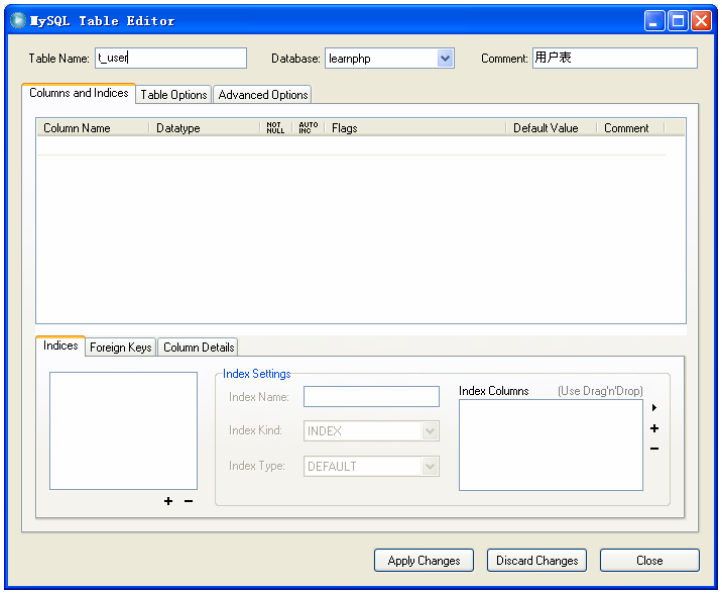


图 8-26　表设计界面

在图 8-26 中，在 Table Name 中输入表名，比如输入“t_user”表示用户表，在 Database 中选择该表所在的数据库（系统已经默认为“Learnphp”，所以不需要修改），在 Comment 里面可以写上对该表的说明。

接下来需要创建表的字段列，在图 8-26 中，双击“Column Name”下面的空白处，会自动显示一个文本框，用于输入字段名，如图 8-27 所示。

可以在设计图中设定表列的类型、默认值、列说明、主键和外键等。

所有列创建完毕后，单击按钮“Apply Changes”，将弹出如图 8-28 所示的界面。

在图 8-28 中，显示了要创建该表的 SQL 语句，单击“Execute”按钮，执行该 SQL 语句。

创建完成后单击“Close”按钮，回到主界面，此时表“t_user”已经创建成功，如图 8-29 所示。

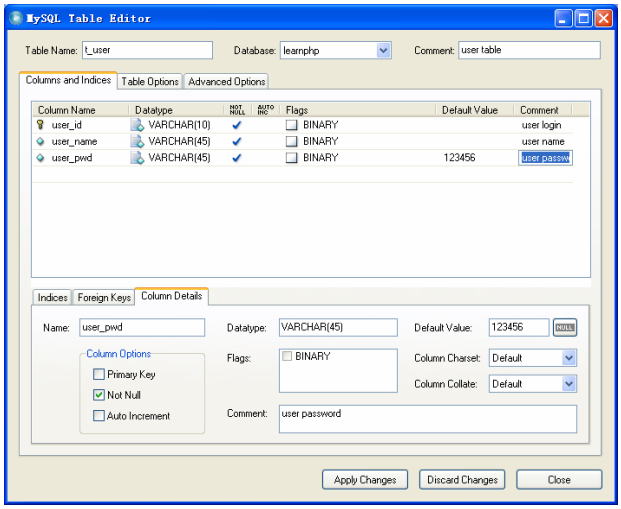


图 8-27　设计数据库表

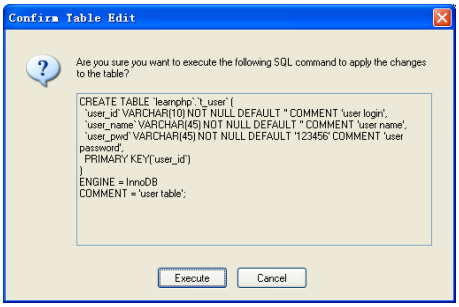


图 8-28　确认是否创建该表

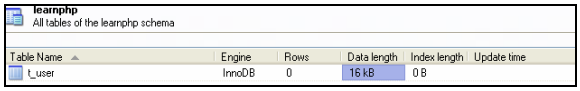


图 8-29　表创建成功

在主界面上，切换到“Stored procedured”属性页，可以创建存储过程。在该属性页单击“Create Stored Proc”按钮，会弹出对话框，让用户输入存储过程名称，如“sp_test”，然后单击“Create PROCEDURE”按钮，会弹出存储过程设计对话框，如图 8-30 所示。

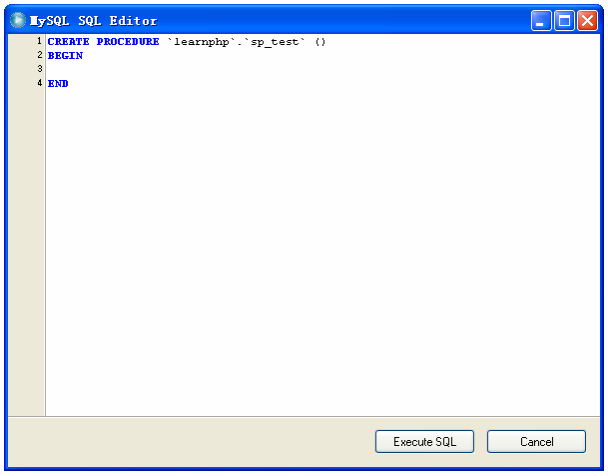


图 8-30　存储过程设计对话框

用户可以根据需要来设计自己的存储过程，创建完成后，单击“Execute SQL”执行。同理，可以切换到“Views”属性页来创建视图。

8.5　使用 phpMyAdmin

除了上面介绍的 MySQL Administrator 之外，还有一个图形化的 MySQL 管理软件。该软件是 B/S 结构的，软件名称为 phpMyAdmin。网上很多地方都可以下载。下载完后，解压缩，将该软件拷贝到 PHP 的根目录下，此处是“d:\masterphp”，拷贝后完整路径为 D:\MasterPHP\phpMyAdmin。打开该目录下的 config.inc.php 文件，需要进行一些设置。

（1）搜索$cfg['PmaAbsoluteUri']，设置 phpmyadmin 的 URL，如“$cfg['PmaAbsoluteUri'] = 'http://localhost/phpmyadmin/';”。

（2）搜索$cfg['blowfish_secret']，设定好 root 密码后这里也要填写，比如 ROOT 密码 123456 则设置“$cfg['blowfish_secret'] = '123456';”。

（3）搜索$cfg['DefaultLang']，将其设置为 zh-gb2312。

（4）搜索$cfg['DefaultCharset']，将其设置为 gb2312。

（5）搜索$cfg['Servers'][$i]['auth_type']，默认为 config，（是不安全的，不推荐），推荐使用 cookie，将其设置为“$cfg['Servers'][$i]['auth_type'] = 'cookie';”。

（6）这里如果设置为 config，要在下面设置用户名和密码。例如：

```
$cfg['Servers'][$i]['user'] = 'root';      // MySQL user-----MySQL 连接用户
$cfg['Servers'][$i]['password'] = '123456';
```

进行完以上修改后，保存 config.inc.php 文件，接着在 IE 浏览器中输入 http://localhost/phpMyAdmin/user_password.php，将出现如图 8-31 所示的界面。

在图中输入登录名称“root”，登录密码“123456”，然后单击“登录”按钮，可以进入到该系统的管理界面，如图 8-32 所示。



图 8-31　phpMyadmin 登录界面

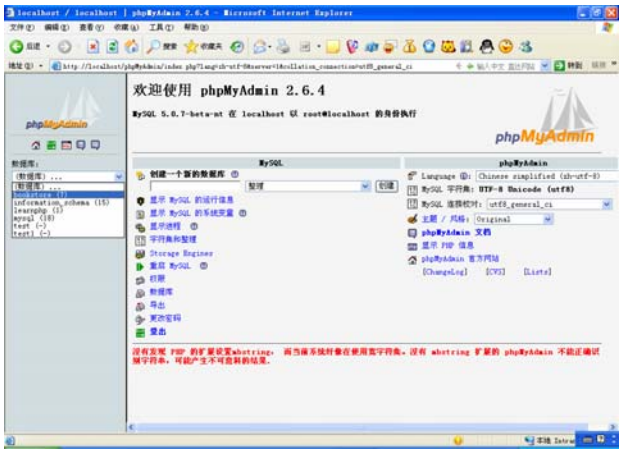


图 8-32　phpMyadmin 管理界面

该软件使用比较简单，限于篇幅，不再详细介绍。

第 9 章　PHP 的数据库编程

第 8 章详细地介绍了 MySQL 数据库的基本知识。本章将介绍如何通过 PHP 连接数据库，并进行各种常见的数据库处理。

9.1　通过 PHP 操作 MySQL 数据库

要进行数据库编程，首先需要连接到数据库。在 PHP 中，提供了大量用于数据库处理的函数，读者可以很方便地进行各种操作。在 PHP 5.0 中，提供了一个新的 MySQL 函数库，即 mysqli（i 表示改进）。这个函数库适合 MySQL 4.0 及以上版本。在 mysqli 函数库中，允许采用面向对象和面向过程的方法来使用。而之前的 PHP 版本采用的是 mysql 函数库，该函数库能够连接 MySQL3.0 以上版本。

为了能够让程序适应 MySQL 的各种版本，本书重点介绍 mysql 函数库。



提示

需要参照第 1 章安装 MySQL 的相关内容，进行如下配置。

在 php.ini 文件中查找“extension=php_mysql.dll”字符串。

原始设置为：

```
;extension=php_mysql.dll
```

将其修改为：

```
extension=php_mysql.dll
```

这样将通过 mysql 函数库来执行，否则不能调用该函数库。

9.1.1　PHP 连接 MySQL

首先介绍如何连接到服务器。

1．连接到服务器

mysql 函数库中提供的连接数据库的函数是：

```
resource mysql_connect ( [string server [, string username [, string password
[, bool new_link [, int client_flags]]]]] )
```

server：MySQL 服务器。可以包括端口号，例如"hostname:port"，或者到本地套接字的路径，例如对于 localhost 的“:/path/to/socket”。如果 PHP 指令 mysql.default_host 未定义（默认情况），则默认值是“localhost:3306”。

username：用户名。默认值是服务器进程所有者的用户名。

password：密码。默认值是空密码。

new_link：如果用同样的参数第 2 次调用 mysql_connect()，将不会建立新连接，而将返回已经打开的连接标识。参数 new_link 改变此行为并使 mysql_connect()总是打开新的连接，甚至当 mysql_connect()曾被用同样的参数调用过时。

client_flags：client_flags 参数可以是以下常量的组合：MYSQL_CLIENT_COMPRESS、MYSQL_CLIENT_IGNORE_SPACE 或 MYSQL_CLIENT_INTERACTIVE。

MYSQL_CLIENT_COMPRESS：使用压缩的通信协议。

MYSQL_CLIENT_IGNORE_SPACE：允许在函数名后留空格位。

MYSQL_CLIENT_INTERACTIVE：允许设置断开连接之前所空闲等候的 interactive_timeout 时间。

通常情况下后面两个参数不使用，只需要前面的 3 个参数就可以了。如要连接前 1 章创建的 bookstore 数据库，可以采用以下语句：

```
$link = mysql_connect("localhost", "root", "123456");
```

如果该函数调用成功，则会返回连接句柄。如果失败，则会返回 FALSE。下面通过一个简单实例说明。

实例 9-1　连接数据库

本实例演示如何通过 mysql_connect 函数连接 mysql 数据库，如代码 9-1 所示。

【代码 9-1】　mysql_connect.php

```
<?php
$link = mysql_connect('localhost', 'root', '123456');
if (!$link) {
   die ('连接失败：' . mysql_error());
}
echo '服务器信息：' .mysql_get_host_info($link);
mysql_close($link);  //关闭连接
?>
```

以上代码连接到本地机器的 MySQL 数据库，如果连接成功，则输入服务器信息。执行结果如图 9-1 所示。

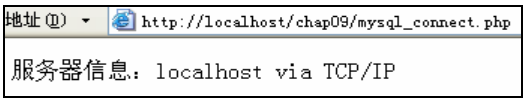


图 9-1　连接 mysql 数据库

mysql_get_host_info($link)函数用于获取服务器信息。mysql_close($link)用于关闭一个活跃的数据库连接。

提示　在 mysqli 函数库中，将使用 mysqli_connect 函数代替 mysql_connect，mysqli_connect 函数的声明如下：

```
mysqli mysqli_connect ( [string host [, string username [, string passwd [, string dbname [, int port [, string socket]]]]]] )
```

2．关闭数据库连接

关闭数据库连接的函数上一节已经使用到了。通常创建一个连接之后，如果不再使用，应该关闭该连接以释放资源，同时可以避免出现各种意外错误。

关闭数据库连接的函数名称为 mysql_close，该函数声明如下所示：

```
bool mysql_close ( [resource link_identifier] )
```

link_identifier：MySQL 的连接标识符，如上一个实例创建的$link。如果没有指定，默认使

用最后被 mysql_connect()打开的连接。如果没有找到该连接，函数会尝试调用 mysql_connect()建立连接并使用它。如果发生意外，没有找到连接或无法建立连接，系统发出 E_WARNING 级别的警告信息。

返回值：如果成功则返回 TRUE，失败则返回 FALSE。

mysql_close()关闭指定的连接标识所关联的到 MySQL 服务器的非持久连接。如果没有指定 link_identifier，则关闭上一个打开的连接。

通常并不强制使用 mysql_close()，因为已打开的非持久连接会在脚本执行完毕后自动关闭。但是建议读者养成良好习惯，在使用完毕后关闭连接。

9.1.2 选择数据库

前面已经成功连接了数据库服务器，但是一个数据库服务器可能包含了很多的数据库。通常需要针对某个具体的数据库进行编程，此时就必须选择某个数据库。

选择目标数据库的函数名称为 mysql_select_db，该函数声明如下：

```
bool mysql_select_db (string database_name [, resource link_identifier] )
```

如果成功则返回 TRUE，失败则返回 FALSE。

mysql_select_db()设定与指定的连接标识符所关联的服务器上的当前激活数据库。如果没有指定连接标识符，则使用上一个打开的连接。如果没有打开的连接，本函数将无参数调用 mysql_connect()尝试打开一个并使用。

实例 9-2 选择数据库

本实例演示如何选择数据库，如代码 9-2 所示。

【代码 9-2】 mysql_select_db.php

```
<?php
$lnk = mysql_connect('localhost', 'root', '123456')
      or die ('连接失败 : ' . mysql_error());

//设定当前的连接数据库为 bookstore
if (mysql_select_db('bookstore', $lnk))
    echo "已经选择数据库 bookstore";
else
    echo ('数据库选择失败 : ' . mysql_error());
?>
```

执行结果图 9-2 所示。



图 9-2 选择数据库

9.1.3 创建查询

进行数据查询的函数是 mysql_query，该函数的声明如下：

```
resource mysql_query ( string query [, resource link_identifier] )
```

其中参数 query 是查询字符串，而 link_identifier 是创建的活动数据库连接，也就是前面通过 mysql_connect 创建的句柄。

mysql_query()向与指定的连接标识符关联的服务器中的当前活动数据库发送一条查询。如果没有指定 link_identifier，则使用上一个打开的连接。如果没有打开的连接，本函数会尝试无参数调用 mysql_connect()函数来建立一个连接并使用之。查询结果会被缓存。



查询字符串不应以分号结束，这和在命令行模式下使用 SQL 语句是有区别的。

mysql_query()仅对 SELECT、SHOW、EXPLAIN 或 DESCRIBE 语句返回一个资源标识符，如果查询执行不正确则返回 FALSE。对于其他类型的 SQL 语句，mysql_query()在执行成功时返回 TRUE，出错时返回 FALSE。非 FALSE 的返回值意味着查询是合法的并能够被服务器执行。这并不说明任何有关影响到的或返回的行数。很有可能一条查询执行成功了，但并未影响到或并未返回任何行。

如果遇到以下情况时，会返回 FALSE：

① 一个用户通过锁定数据库以独占方式访问该表；

② MySQL 服务器不能再执行更多的 SQL 查询；

③ 查询语句中用到的表不存在；

④ 请求信息的用户不具备所需要的权限。

假设查询成功，可以调用 mysql_num_rows()来查看对应的 SELECT 语句返回了多少行，或者调用 mysql_affected_rows()来查看对应的 DELETE、INSERT、REPLACE 或 UPDATE 语句影响到了多少行。

仅对 SELECT、SHOW、DESCRIBE 或 EXPLAIN 语句，mysql_query()才会返回一个新的结果标识符，可以将其传递给 mysql_fetch_array()和其他处理结果表的函数。处理完结果集后可以通过调用 mysql_free_result()来释放与之关联的资源，尽管脚本执行完毕后会自动释放内存。

实例 9-3　查询数据库表

本实例演示如何在 PHP 中通过 SELECT 语句查询记录，如代码 9-3 所示。

【代码 9-3】　mysql_query.php

```
<?php
$lnk = mysql_connect('localhost', 'root', '123456')
      or die ('连接失败 : ' . mysql_error());

//设定当前的连接数据库为 bookstore
if (mysql_select_db('bookstore', $lnk))
    echo "已经选择数据库 bookstore<br>";
else
    echo ('数据库选择失败 : ' . mysql_error());
```

```
$result = mysql_query("SELECT * from aaa",$lnk)  //设定$lnk
   or echo("查询表 aaa 失败: " . mysql_error());

//可以不设定$lnk，默认为最近创建的连接
$result = mysql_query("SELECT * from items")
    or echo("<br>查询表 items 失败: " . mysql_error());
?>
```

输出结果如图 9-3 所示。



图 9-3　查询数据库表

以上代码首先创建了一个数据库连接，然后选择数据库 bookstore，接着从表 aaa 中查询数据，由于表 aaa 不存在，所以输出查询失败。函数 mysql_error()显示错误原因。最后创建一个从 items 表中的查询，由于该表存在，所以创建成功，没有输出错误提示。

9.1.4　显示查询结果

前面介绍了如何在 PHP 中创建查询，接下来介绍如何检索被查询的数据。PHP 提供了两种方式来进行数据的检索。

（1）mysql_fetch_row 函数。mysql_fetch_row 函数，从结果集中取得一行作为枚举数组，该函数声明如下：

```
array mysql_fetch_row (resource result )
```

返回根据所取得的行生成的数组，如果没有更多行则返回 FALSE。mysql_fetch_row()从和指定的结果标识关联的结果集中取得一行数据并作为数组返回。每个结果的列存储在一个数组的单元中，偏移量从 0 开始。依次调用 mysql_fetch_row()将返回结果集中的下一行，如果没有更多行则返回 FALSE。

实例 9-4　检索查询记录

本实例演示如何从数据库表中检索记录并显示，如代码 9-4 所示。

【代码 9-4】　mysql_fetch_row.php

```
<?php
…省略部分数据库连接代码
$result = mysql_query("SELECT name,category_id from categories")
    or die("<br>查询表 categories 失败: " . mysql_error());

//创建记录集
$row=mysql_fetch_row($result);
while ($row)
{
    echo $row[0].'--'.$row[1] . "<br>";
```

```
    $row=mysql_fetch_row($result);
}
?>
```

执行结果如图 9-4 所示。



图 9-4　检索查询记录

在代码中，记录集直接赋值给变量$row，然后通过$row['字段索引']来获取数据。通过循环获取数据，直到$row 返回 FALSE。其中字段索引是按照 SELECT 出现的先后次序来确定的，第一个字段的索引为 0。

在书写查询语句的时候，也可以不指定字段名称，直接通过“*”来查询，这样在显示结果的时候，如果按照字段索引来获取数据，则是按照数据库创建时的默认次序显示。

实例 9-5　获取表的全部字段

本实例通过“Select * from”语句查询所有的表字段信息，如代码 9-5 所示。

【代码 9-5】　mysql_fetch_row_1.php

```
<?php
…省略部分数据库连接代码
$result = mysql_query("SELECT * from categories")
    or die("<br>查询表 categories 失败: " . mysql_error());

//创建记录集
$row=mysql_fetch_row($result);
while ($row)
{
    echo $row[0].'--'.$row[1] . "<br>";
    $row=mysql_fetch_row($result);
}
//mysql_free_result($result); 该句代码可以不要，用于释放资源
?>
```

执行结果如图 9-5 所示。



图 9-5　通过“select * from”语句查询记录

在这段代码中，查询语句查询所有字段名称，如果不清楚数据库中字段的明确顺序，最好通过字段名称来输出结果。因此如果想要按照指定的顺序输出，最好在 Select 语句中指定检索顺序。需要注意的是，不通过$row['字段名称']来获取数据，但是后面介绍的 mysql_fetch_array 和 mysql_fetch_assoc 函数可以。

（2）mysql_fetch_array 函数。该函数从结果集中取得一行作为关联数组，或数字数组，或二者兼有，该函数声明如下：

```
array mysql_fetch_array ( resource result [, int result_type] )
```

返回根据从结果集取得的行生成的数组，如果没有更多行则返回 FALSE。mysql_fetch_array()是 mysql_fetch_row()的扩展版本。除了将数据以数字索引方式存储在数组中之外，还可以将数据作为关联索引存储，用字段名作为键名。

如果结果中的两个或以上的列具有相同字段名，最后一列将优先。要访问同名的其他列，必须用该列的数字索引或给该列起个别名。对有别名的列，不能再用原来的列名访问其内容，如下所示，为具有相同列名的字段创建别名：

```
select table1.field as foo, table2.field as bar from table1, table2
```

mysql_fetch_array()中可选的第二个参数 result_type 是一个常量，可以接受以下值：MYSQL_ASSOC，MYSQL_NUM 和 MYSQL_BOTH。本参数的默认值是 MYSQL_BOTH。

如果用了 MYSQL_BOTH，将得到一个同时包含关联和数字索引的数组。用 MYSQL_ASSOC 只得到关联索引（如同 mysql_fetch_assoc()那样，只能通过字段名称），用 MYSQL_NUM 只得到数字索引（如同 mysql_fetch_row()那样）。

提示 本函数返回的字段名是区分大小写的。

实例 9-6 通过 mysql_fetch_array 函数获取记录

本实例演示通过 mysql_fetch_array 函数获取查询记录，如代码 9-6 所示。

【代码 9-6】 mysql_fetch_array.php

```
<?php
…省略部分数据库连接代码
$result = mysql_query("SELECT * from members")
    or die("<br>查询表 members 失败: " . mysql_error());

//创建记录集
$row=mysql_fetch_array($result);
while ($row)
{
    echo $row[0].'--'.$row['1'] . '--'.$row['email'] ."<br>";
    $row=mysql_fetch_array($result);
}
?>
```

执行结果如图 9-6 所示。

在以上代码中，既可以使用字段名称$row['email']，也可以通过字段索引$row['1']来检索数据。

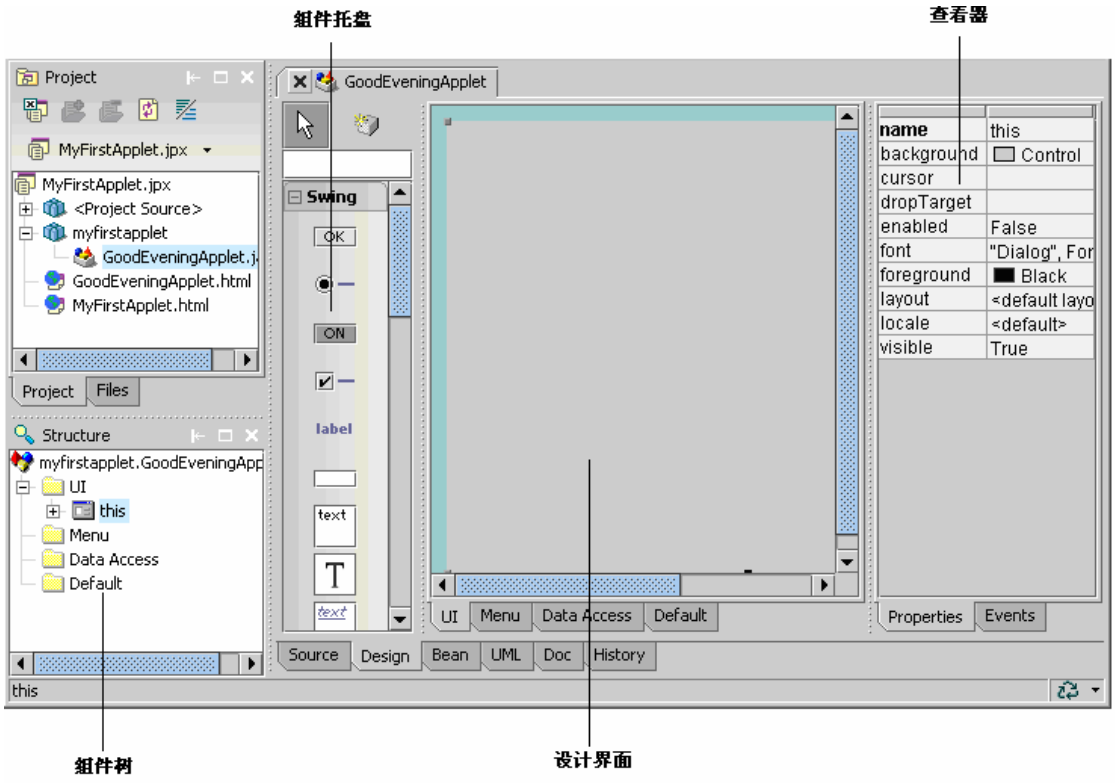


图 9-6　mysql_fetch_array 获取记录

实例 9-7　通过字段别名获取记录

本实例演示通过字段别名来获取记录，如代码 9-7 所示。

【代码 9-7】　mysql_fetch_array_1.php

```
<?php
…省略部分数据库连接代码
$result = mysql_query("SELECT categories.name as categroy_name,
    items.name as book_name from categories,items
    where categories.category_id=items.category_id and
    items.item_id=5")
    or die("<br>查询表 items 失败: " . mysql_error());

//创建记录集
$row=mysql_fetch_array($result);
while ($row)
{
    echo $row['categroy_name'].'--'.$row['book_name'] ."<br>";
    $row=mysql_fetch_array($result);
}
?>
```

执行结果如图 9-7 所示。



图 9-7 通过字段索引获取记录

以上代码从两个表中检索的列列名相同，因此采用了别名。采用别名后，不能使用原来的列名。但采用别名，仍然可以使用字段索引检索。

（3）mysql_fetch_assoc 函数。该函数同 mysql_fetch_row 的功能基本类似，但是用法上稍有区别。该函数声明如下：

```
array mysql_fetch_assoc ( resource result )
```

返回根据从结果集取得的行生成的关联数组，如果没有更多行则返回 FALSE。

mysql_fetch_assoc()和用 mysql_fetch_array()加上第二个可选参数 MYSQL_ASSOC 完全相同，仅仅返回关联数组。这也是 mysql_fetch_array()起始的工作方式。如果在关联索引之外还需要数字索引，用 mysql_fetch_array()。

如果结果中的两个或以上的列具有相同字段名，最后一列将优先。要访问同名的其他列，可以用 mysql_fetch_row()来取得数字索引或给该列起个别名。

提示 本函数返回的字段名是区分大小写的。

实例 9-8 通过 mysql_fetch_assoc 函数获取记录

本实例演示通过 mysql_fetch_assoc 函数获取记录，如代码 9-8 所示。

【代码 9-8】 mysql_fetch_assoc.php

```
<?php
…省略部分数据库连接代码
$result = mysql_query("SELECT * from members")
    or die("<br>查询表 members 失败: " . mysql_error());

//创建记录集
$row=mysql_fetch_assoc($result);
while ($row)
{
    echo $row['member_id'].'--'.$row['member_login'] . "<br>";
    $row=mysql_fetch_assoc($result);
}
?>
```

执行结果如图 9-8 所示。



图 9-8 mysql_fetch_assoc 函数获取记录

（4）获取检索记录数量的函数 mysql_num_rows。前面的 3 个函数用于检索记录，但是并没有直观地反映出一共检索了多少条记录。mysql_num_rows 函数可以统计出获得的记录数量。该函数声明如下：

```
int mysql_num_rows ( resource result )
```

mysql_num_rows()返回结果集中行的数目。此命令仅对 SELECT 语句有效。要取得被 INSERT、UPDATE 或者 DELETE 查询所影响到的行的数目，要用 mysql_affected_rows()。



如果使用 mysql_unbuffered_query()，则直到结果集中的所有行都被提取后，mysql_num_rows()才能返回正确的值。

实例 9-9　获取被查询的记录数目

本实例演示如何获取被查询的记录数目，如代码 9-9 所示。

【代码 9-9】　mysql_num_rows.php

```
<?php
…省略部分数据库连接代码
$result = mysql_query("select * from orders")
    or die("<br>查询表 orders 失败: " . mysql_error());
$rows=mysql_num_rows($result);  //取得记录数量
echo "总记录数:  $rows <br>";
echo "<table border=1><tr><td>订单编号</ td>";
echo "<td>会员编号</ td><td>产品编号</td><td>数量</td></tr>";
for ($i=0;$i<$rows;$i++)  //这里同前面不一样，因为已经知道记录数量，采用 for 循环
{
    $row=mysql_fetch_array($result);

    echo "<tr><td> $row[0] </td> <td> $row[1] </td>";
    echo "<td> $row[2] </td><td> $row[3] </td></tr>";
}
echo "</table>";
?>
```

执行结果如图 9-9 所示。

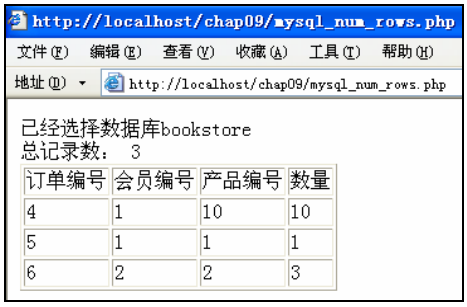


图 9-9　获取被查询的记录数量

以上代码首先获得总的记录数量，然后通过 for 循环输出。



在 MySQL 数据库一章中介绍的各种 SELECT 语句，都能够通过上面介绍的函数来实现，包括统计函数等。

实例 9-10　通过 Count 语句获取被查询的记录数目

本实例演示通过 count 语句统计被查询的记录数目，如代码 9-10 所示。

【代码 9-10】　mysql_fetch_array_2.php

```
    <?php
    …省略部分数据库连接代码
    $result = mysql_query("select count(*) as order_num,sum(quantity) as total
from orders")
        or die("<br>查询表 orders 失败: " . mysql_error());

    echo "<table border=1><tr><td>总订单数</ td>";
    echo "<td>总定购数量</td></tr>";
    $row=mysql_fetch_array($result);

    echo "<tr><td>". $row['order_num'] ."</td> <td>". $row['total']."
</td></tr>";
    echo "</table>";
    ?>
```

执行结果如图 9-10 所示。



图 9-10　通过 count 语句统计查询记录数

9.1.5　插入新数据

前面介绍了如何从数据库中检索数据，接下来介绍如何向数据库中插入新数据。插入新数据同检索数据在用法上没有本质区别，惟一不同的是使用不同的 SQL 语句。

要插入一个新数据，同检索一样，首先要连接到 MySQL 服务器，然后选择一个数据库，最后创建要执行的 SQL 语句。前面章节已经介绍了插入数据的 INSERT 语句，接下来通过实例介绍如何向数据库中添加记录。

实例 9-11　向表中插入数据

本实例演示了如何向数据库表中插入数据，如代码 9-11 所示。

【代码 9-11】　insert.php

```
    <?php
    $lnk = mysql_connect('localhost', 'root', '123456')
          or die ('连接失败 : ' . mysql_error());

    //设定当前的连接数据库为 bookstore
```

```
if (mysql_select_db('bookstore', $lnk))
    echo "已经选择数据库 bookstore<br>";
else
    echo ('数据库选择失败 : ' . mysql_error());

$myquery="insert into orders (member_id,item_id,quantity)
 values (1,5,10)";  //设定插入语句

$result=mysql_query($myquery)
    or die("<br>插入失败: " . mysql_error());;  //执行插入 sql 语句

//检索判断是否新数据已经插入成功
$result = mysql_query("select * from orders")
    or die("<br>查询表 orders 失败: " . mysql_error());
$rows=mysql_num_rows($result);  //取得记录数量
echo "总记录数:  $rows <br>";
echo "<table border=1><tr><td>订单编号</ td>";
echo "<td>会员编号</ td><td>产品编号</td><td>数量</td></tr>";
for ($i=0;$i<$rows;$i++)
{
    $row=mysql_fetch_array($result);
    echo "<tr><td> $row[0] </td> <td> $row[1] </td>";
    echo "<td> $row[2] </td><td> $row[3] </td></tr>";
}
echo "</table>";
echo "记录插入成功";
?>
```

执行结果如图 9-11 所示。



图 9-11　插入记录

9.1.6　编辑数据

编辑数据同插入数据类似，不同的是使用 UPDATE 关键字来修改记录。

实例 9-12　编辑数据

本实例演示如何在 PHP 中编辑数据库中的记录，如代码 9-12 所示。

【代码 9-12】　update.php

```
<?php
```

```
    …省略部分数据库连接代码
    $myquery="update orders set quantity=100";  //设定 sql 语句，该语句将所有数量全部变成 100
    $result=mysql_query($myquery)
        or die("<br>编辑失败: " . mysql_error());;  //执行修改 sql 语句
    //检索判断是否新数据已经修改成功
    $result = mysql_query("select * from orders")
        or die("<br>查询表 orders 失败: " . mysql_error());
    ows=mysql_num_rows($result);  //取得记录数量
    echo "总记录数:  $rows <br>";
    echo "<table border=1><tr><td>订单编号</ td>";
    echo "<td>会员编号</ td><td>产品编号</td><td>数量</td></tr>";
    for ($i=0;$i<$rows;$i++)
    {
        $row=mysql_fetch_array($result);

        echo "<tr><td> $row[0] </td> <td> $row[1] </td>";
        echo "<td> $row[2] </td><td> $row[3] </td></tr>";
    }
    echo "</table>";
    echo "记录编辑成功";
    ?>
```

执行结果如图 9-12 所示。



图 9-12　编辑记录

9.1.7　删除数据

删除数据是通过 DELETE 关键字来实现的，过程同 UPDATE 一样。

实例 9-13　删除记录

本实例演示如何在 PHP 中删除记录，如代码 9-13 所示。

【代码 9-13】　delete.php

```
    <?php
    …省略部分数据库连接代码
    $myquery="delete from orders where order_id=6";  //设定 sql 语句，该语句将所有数量全部变成 100，如果 order_id 为 6 的记录不存在，则删除记录数量为 0
```

```
$result=mysql_query($myquery)
    or die("<br>删除失败: " . mysql_error());;  //执行删除 sql 语句
//获取被删除记录数量
echo "被删除记录数量为: " . mysql_affected_rows($lnk);
//检索判断是否新数据已经删除成功
$result = mysql_query("select * from orders")
    or die("<br>查询表 orders 失败: " . mysql_error());
$rows=mysql_num_rows($result);  //取得记录数量
echo "总记录数:  $rows <br>";
echo "<table border=1><tr><td>订单编号</ td>";
echo "<td>会员编号</ td><td>产品编号</td><td>数量</td></tr>";
for ($i=0;$i<$rows;$i++)
{
    $row=mysql_fetch_array($result);
    echo "<tr><td> $row[0] </td> <td> $row[1] </td>";
    echo "<td> $row[2] </td><td> $row[3] </td></tr>";
}
echo "</table>";
echo "记录编辑成功";
?>
```

执行结果如图 9-13 所示。

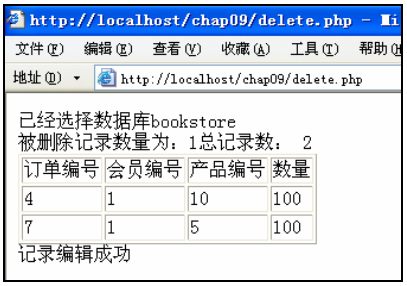


图 9-13　删除记录

9.1.8　获取被影响的记录行数

在 delete.php 代码中使用了一个 mysql_affected_rows($lnk)函数，该函数可以用来判断一次操作，即取得前一次 MySQL 操作所影响的记录行数，除了可以应用在 DELETE 中，还可以应用到 INSERT、UPDATE 语句中。该函数声明如下：

```
int mysql_affected_rows ( [resource link_identifier] )
```

该函数取得最近一次与 link_identifier 关联的 INSERT，UPDATE 或 DELETE 查询所影响的记录行数。

link_identifier：MySQL 的连接标识符。如果没有指定，默认使用最后被 mysql_connect()打开的连接。如果没有找到该连接，函数会尝试调用 mysql_connect()建立连接并使用它。如果发生意外，没有找到连接或无法建立连接，系统发出 E_WARNING 级别的警告信息。

执行成功则返回受影响的行的数目，如果最近一次查询失败，函数返回–1。

如果最近一次操作是没有任何条件（WHERE）的 DELETE 查询，在表中所有的记录都会被删除。

当使用 UPDATE 查询，MySQL 不会将原值和新值一样的列更新。这样 mysql_affected_rows()函数返回值不一定就是查询条件所符合的记录数，只有真正被修改的记录数才会被返回。

REPLACE 语句首先删除具有相同主键的记录，然后插入一个新记录。本函数返回的是被删除的记录数加上被插入的记录数。

9.1.9 创建数据库

前面介绍了如何操纵数据库表中的数据，接下来介绍如何在 PHP 中创建数据库、数据库表等高级应用。

在 PHP 中，创建数据库使用函数 mysql_create_db，该函数的声明如下：

```
bool mysql_create_db ( string database name [, resource link_identifier] )
```

mysql_create_db()尝试在指定的连接标识所关联的服务器上建立一个新数据库。如果成功则返回 TRUE，失败则返回 FALSE。但是通过该方法创建的数据库效率比较低，建议直接使用 SQL 语句来创建数据库。

实例 9-14 创建数据库

本实例演示如何在 PHP 中创建数据库，如代码 9-14 所示。

【代码 9-14】 createdatabase.php

```
<?php
$lnk = mysql_connect('localhost', 'root', '123456')
      or die ('连接失败 : ' . mysql_error());
$myquery="create database test3";  //创建数据库 test3
$result=mysql_query($myquery)
    or die("<br>插入失败: " . mysql_error());;  //执行创建数据库 sql 语句
echo "数据库创建成功!";
?>
```

在 MySQL 客户端可以查看新创建的数据库，如图 9-14 所示。

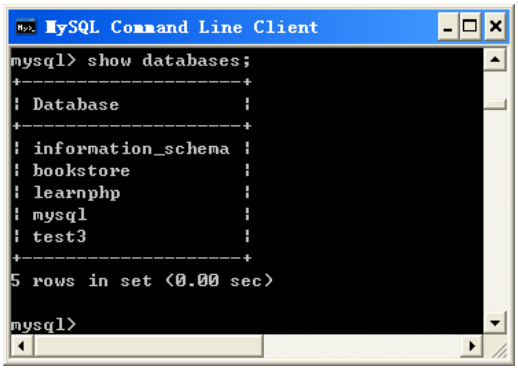


图 9-14 新创建的数据库

9.1.10 创建数据库表

创建数据库表将使用 CREATE TABLE 关键字。

实例 9-15　创建数据库表

本实例演示如何在 PHP 中创建数据库表，如代码 9-14 所示。

【代码 9-15】　createtable.php

```
<?php
$lnk = mysql_connect('localhost', 'root', '123456')
      or die ('连接失败 : ' . mysql_error());
//设定当前的连接数据库为 test3
if (mysql_select_db('test3', $lnk))
    echo "已经选择数据库 test3<br>";
else
    echo ('数据库选择失败 : ' . mysql_error());
$myquery="create table if not exists t_user (name varchar(20) not null ,
       pwd  varchar(10))";  //创建数据库表
$result=mysql_query($myquery)
    or die("<br>数据库表 t_user 创建失败: " . mysql_error());;  //执行 sql 语句

echo "数据库表 t_user 创建成功!";
?>
```

执行以上代码之后，可以在 MySQL 客户端查看是否已经创建成功，如图 9-15 所示。

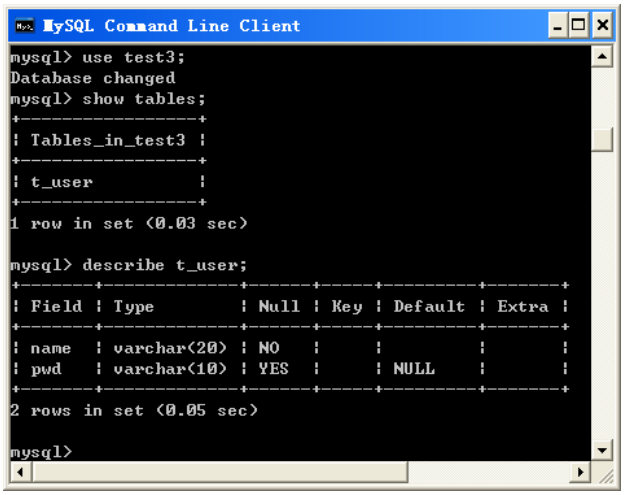


图 9-15　创建数据库表

由图 9-15 可以看出，表 t_user 已经创建成功。

9.1.11　事务处理

事务处理在第 8 章已经介绍了，下面介绍在 PHP 中如何进行事务处理。执行过程同在 MySQL 客户端基本类似，主要是调用 SQL 语句来执行。

实例 9-16　事务处理

本实例演示如何在 PHP 中进行事务处理，如代码 9-16 所示。

【代码 9-16】 transaction.php

```
<?php
$lnk = mysql_connect('localhost', 'root', '123456')
      or die ('连接失败 : ' . mysql_error());
//设定当前的连接数据库为 bookstore
if (mysql_select_db('bookstore', $lnk))
    echo "已经选择数据库 bookstore<br>";
else
    echo ('数据库选择失败 : ' . mysql_error());
$myquery="start transaction";  //启动事务
$result=mysql_query($myquery)
    or die("<br>事务启动失败: " . mysql_error());  //执行启动事务 sql 语句
$myquery="insert into orders (member_id,item_id,quantity)
 values (1,5,1000)";  //设定插入语句

$result=mysql_query($myquery)
    or die("<br>插入失败: " . mysql_error());  //执行插入 sql 语句

$myquery="insert into orders (member_id,item_id,quantity)
 values (2,5,1000)";  //设定插入语句

$result=mysql_query($myquery)
    or die("<br>插入失败: " . mysql_error()); //执行插入 sql 语句

$myquery="commit";  //提交事务
echo "事务提交!<br>";
echo "(1,5,1000)插入成功!<br>";
echo "(2,5,1000)插入成功!<br>";

$result=mysql_query($myquery)
    or die("<br>事务提交失败: " . mysql_error());  //提交事务

//再次启动事务，这次由于某种些原因，导致失败，采用 rollback 取消事务
$myquery="start transaction";  //启动事务

$result=mysql_query($myquery)
    or die("<br>事务启动失败: " . mysql_error());  //执行启动事务 sql 语句

$myquery="insert into orders (member_id,item_id,quantity)
 values (1,5,999)";  //设定插入语句

$result=mysql_query($myquery)
    or die("<br>插入失败: " . mysql_error());  //执行插入 sql 语句;
echo "(1,5,999)插入成功，单事务还未提交!<br>";

$myquery="insert into orderss (member_id,item_id,quantity)
 values (2,5,999)";  //设定插入语句，由于 orderss 表不存在，导致错误

if (!$result=mysql_query($myquery))
{
```

```
        mysql_query("rollback");  //调用 rollback 取消事务，这样会导致前面的一条记录插入也失败

        echo "(2,5,999)插入失败因为 orderss 表不存在!<br>";
        echo "事务被取消,(1,5,999)和(2,5,999)的数据插入均被取消!<br>";
    }
    else
    {
        mysql_query("commit");
    }

    //检索判断是否新数据已经插入成功
    $result = mysql_query("select * from orders")
        or die("<br>查询表 items 失败: " . mysql_error());
    $rows=mysql_num_rows($result);  //取得记录数量
    echo "总记录数:  $rows <br>";
    echo "<table border=1><tr><td>订单编号</ td>";
    echo "<td>会员编号</ td><td>产品编号</td><td>数量</td></tr>";
    for ($i=0;$i<$rows;$i++)  //
    {
        $row=mysql_fetch_array($result);

        echo "<tr><td> $row[0] </td> <td> $row[1] </td>";
        echo "<td> $row[2] </td><td> $row[3] </td></tr>";
    }
    echo "</table>";
    ?>
```

执行结果如图 9-16 所示。

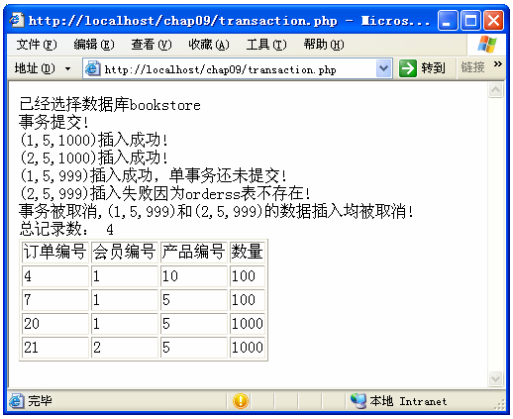


图 9-16　事务处理

9.1.12　存储过程

下面介绍如何在 PHP 中调用在 MySQL 中创建的存储过程，其执行与事务处理类似。

实例 9-17　调用存储过程

本实例演示如何在 PHP 中调用存储过程，如代码 9-17 所示。

【代码 9-17】 stored proc.php

```
<?php
$lnk = mysql_connect('localhost', 'root', '123456')
      or die ('连接失败 : ' . mysql_error());

//设定当前的连接数据库为 bookstore
if (mysql_select_db('bookstore', $lnk))
    echo "已经选择数据库 bookstore<br>";
else
    echo ('数据库选择失败 : ' . mysql_error());

$myquery="call  simpleproc(@a)";  //调用存储过程 simpleproc(@a)

//执行存储过程
$result = mysql_query($myquery);
$result = mysql_query("select @a");  //获取结果集
$row=mysql_fetch_array($result);
echo $row[0];   //获取存储过程的结果
?>
```

执行结果如图 9-17 所示。

以上代码的执行同在 MySQL 客户端执行的结果一样。如果存储过程没有输出参数，则不需要打印结果，直接调用存储过程即可。如果有输出参数，则可以先调用 select 创建结果集，然后调用 mysql_fetch_array 获取结果。



图 9-17 在 PHP 中调用存储过程

9.1.13 其他重要函数

在介绍了前面的一些基本操作之后，读者已经能够进行常用的数据库操作。接下来介绍一些辅助的函数，能够帮助提高编程的灵活性以及编程效率。

（1）mysql_data_seek()函数。该函数移动内部结果的指针，声明如下：

```
bool mysql_data_seek ( resource result_identifier, int row_number )
```

如果成功则返回 TRUE，失败则返回 FALSE。mysql_data_seek()将指定的结果标识所关联的 MySQL 结果内部的行指针移动到指定的行号。接着调用 mysql_fetch_row()将返回那一行。

row_number 从 0 开始。row_number 的取值范围从 0 到 mysql_num_rows−1。



mysql_data_seek()只能和 mysql_query()结合起来使用，而不能用于 mysql_unbuffered_query()。

实例 9-18　在记录中移动指针

本实例演示了如何在 PHP 中移动记录集指针，如代码 9-18 所示。

【代码 9-18】　mysql_data_seek.php

```
<?php
    $link = mysql_pconnect("localhost", "root", "123456")
        or die("Could not connect: " . mysql_error());

    mysql_select_db("bookstore")
        or die("Could not select database: " . mysql_error());

    $query = "SELECT * FROM orders";
    $result = mysql_query($query)
        or die("Query failed: " . mysql_error());

    /* 倒序输出记录 */
    for ($i = mysql_num_rows($result) - 1; $i >= 0; $i--) {  //循环输出
        if (!mysql_data_seek($result, $i)) {
            echo "Cannot seek to row $i: " . mysql_error(). "\n";
            continue;
        }

        if(!($row = mysql_fetch_object($result)))
            continue;

   echo "$row->order_id $row->member_id $row->item_id $row->quantity<br
/>\n";
    }

    mysql_free_result($result);
?>
```

执行结果如图 9-18 所示。



图 9-18　移动记录集

第 1 列数字是 order_id，正常情况下应该是 4，然后是 7，因为在代码中进行了倒序输出，所以先显示 7。

（2）mysql_list_dbs()函数。该函数列出 MySQL 服务器中所有的数据库，声明如下：

```
resource mysql_list_dbs ( [resource link_identifier] )
```

mysql_list_dbs()将返回一个结果指针，包含了当前 MySQL 进程中所有可用的数据库。可以用 mysql_tablename()函数来遍历此结果指针，或者任何使用结果表的函数，例如 mysql_fetch_array()。

实例 9-19　列出所有数据库名称

本实例演示如何获取 mysql 服务器中所有的数据库，如代码 9-19。

【代码 9-19】　mysql_list_dbs.php

```
<?php
$link = mysql_connect('localhost', 'root', '123456');
$db_list = mysql_list_dbs($link);

while ($row = mysql_fetch_object($db_list)) {
    echo $row->Database . "<br>";
}
?>
```

执行结果如图 9-19 所示。



图 9-19　列出所有数据库

提示　以上代码使用了 mysql_fetch_object 函数，它同 mysql_fetch_array 的区别是 mysql_fetch_object 函数返回的是一个对象而不是一个数组。

（3）mysql_field_name()。该函数取得结果中指定字段的字段名，声明如下：

```
string mysql_field_name ( resource result, int field_index )
```

mysql_field_name()返回指定字段索引的字段名。result 必须是一个合法的结果标识符，field_index 是该字段的数字偏移量。field_index 从 0 开始，例如第 3 个字段的索引值是 2，第 4 个字段的索引值是 3，以此类推。

实例 9-20　获取字段名称

本实例演示如何获取指定表格的字段名称，如代码 9-20 所示。

【代码 9-20】　mysql_field_name.php

```
<?php
…省略部分数据库连接代码
$result = mysql_query("select * from orders")
    or die("<br>查询表 orders 失败: " . mysql_error());

echo mysql_field_name($result, 0) ."<br>";
echo mysql_field_name($result, 1) ."<br>";
echo mysql_field_name($result, 2) ."<br>";
echo mysql_field_name($result, 3) ."<br>";
?>
```

执行结果如图 9-20 所示。



图 9-20　获取字段名称

（4）mysql_field_len()函数。该函数返回指定字段的长度，函数声明如下：

```
int mysql_field_len ( resource result, int field_offset )
```

实例 9-21　获取字段长度

本实例演示如何获取字段的长度，如代码 9-21 所示。

【代码 9-21】　mysql_field_len.php

```
<?php
…省略部分数据库连接代码
$result = mysql_query("select * from items")
    or die("<br>查询表 orders 失败: " . mysql_error());
echo mysql_field_len($result, 0) ."<br>";
echo mysql_field_len($result, 1) ."<br>";
echo mysql_field_len($result, 2) ."<br>";
echo mysql_field_len($result, 3) ."<br>";
?>
```

执行结果如图 9-21 所示。



图 9-21　获取字段长度

（5）mysql_field_type()函数。该函数取得结果集中指定字段的类型，声明如下：

```
string mysql_field_type ( resource result, int field_offset )
```

mysql_field_type()和 mysql_field_name()函数相似。参数完全相同，但返回的是字段类型而不是字段名。字段类型有“int”、“real”、“string”、“blob”以及其他。

实例 9-22　获取字段类型

本实例演示如何获取字段类型，如代码 9-22 所示。

【代码 9-22】　mysql_field_type.php

```
<?php
   mysql_connect("localhost", "root", "123456");
   mysql_select_db("bookstore");
```

```
    $result = mysql_query("SELECT * FROM orders"); //查询 orders 表
    $fields = mysql_num_fields($result);  //获取字段数量
    $rows   = mysql_num_rows($result);  //获取记录数量
    $table = mysql_field_table($result, 0);  //获取字段所在的表名
    echo "表名为: '".$table."' 该表有".$fields." 个字段和".$rows."条记录<br>";
    echo "该表有以下字段:<br>";
    for ($i=0; $i < $fields; $i++) {
        $type  = mysql_field_type($result, $i);
        $name  = mysql_field_name($result, $i);
        $len   = mysql_field_len($result, $i);
        $flags = mysql_field_flags($result, $i);
        echo $type." ".$name." ".$len." ".$flags."<br>";
    }
    mysql_free_result($result);
    mysql_close();
?>
```

执行结果如图 9-22 所示。

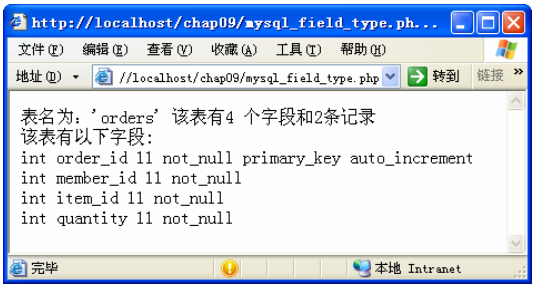


图 9-22　获取字段类型及相关信息



以上代码除使用了 mysql_field_type 函数以外，还使用了 mysql_field_table 函数，该函数用于获取字段所在的表名。mysql_field_flags 函数用于获取某个字段的属性，如是否为空、是否为主键、是否是自动编号等。mysql_num_fields 函数用于取得检索获得的字段数量。

（6）mysql_list_fields 函数。该函数列出 MySQL 数据库表中的字段，声明如下：

```
    resource mysql_list_fields (string database_name, string table_name [,
resource link_identifier] )
```

mysql_list_fields()取得给定表名的信息。参数是数据库名和表名。返回一个结果指针，可以用于 mysql_field_flags()、mysql_field_len()、mysql_field_name()和 mysql_field_type()。

实例 9-23　使用 mysql_list_fields 函数获取表格字段信息

本实例演示如何通过 mysql_list_fields 函数获取数据库表的字段信息，如代码 9-23 所示。

【代码 9-23】　mysql_list_fields.php

```
    <?php
    $link = mysql_connect('localhost', 'root', '123456');
```

```
$fields = mysql_list_fields("bookstore", "orders", $link);
$columns = mysql_num_fields($fields);

for ($i = 0; $i < $columns; $i++) {
    echo mysql_field_name($fields, $i) . "<br>";
}
```

执行结果如图 9-23 所示。



图 9-23　获取字段信息

（7）获取各种信息的函数

mysql_get_client_info 函数用于取得 MySQL 客户端信息，声明如下：

```
string mysql_get_client_info ( void )
```

mysql_get_host_info 函数用于取得 MySQL 主机信息，如下所示：

```
string mysql_get_host_info ( [resource link_identifier] )
```

mysql_get_proto_info 函数用于取得 MySQL 协议信息，声明如下：

```
int mysql_get_proto_info ( [resource link_identifier] )
```

mysql_get_server_info 函数用于取得 MySQL 服务器信息，声明如下：

```
string mysql_get_server_info ( [resource link_identifier] )
```

实例 9-24　获取 mysql 服务器的各种信息

本实例演示如何获取服务器的相关信息，如代码 9-24 所示。

【代码 9-24】　get_info.php

```
<?php
    //获取客户端信息
     printf ("MySQL 客户端信息: %s<br>", mysql_get_client_info());
     //获取主机信息
    mysql_connect("localhost", "root", "123456") or
       die("Could not connect: " . mysql_error());
    printf ("MySQL 主机信息: %s<br>", mysql_get_host_info());

     //获取 mysql 协议信息
    printf ("MySQL 协议版本: %s<br>", mysql_get_proto_info());

     //获取 mysql 服务器版本信息
    printf ("MySQL 服务器版本: %s<br>", mysql_get_server_info());
?>
```

执行结果如图 9-24 所示。

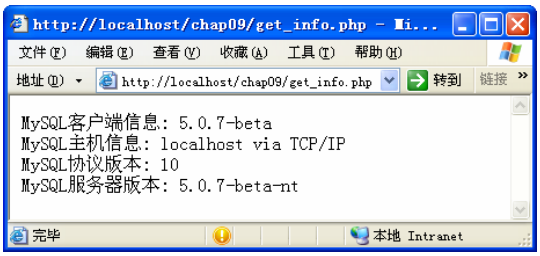


图 9-24　获取服务器信息

9.2　mysqli 函数库应用

mysqli 函数库同 mysql 函数库的应用基本类似，而且大部分的函数使用方法都一样，惟一的区别就是 mysqli 函数库中的函数名称是以 mysqli 开始的，如连接服务器的函数是用 mysqli_connect。

在使用 mysqli 之前，需要在 php.ini 文件中增加一项，即“extension=php_mysqli.dll”。然后关闭 Apache 服务器，并重新启动，这样就可以调用 mysqli 函数库。修改后的配置如图 9-25 所示。

```
;extension=php_mysql.dll
extension=php_mysqli.dll
;extension=php_oci8.dll
```

图 9-25　修改 php.ini 的配置

PHP 5.0 中允许使用面向过程和面向对象两种方法来使用 mysqli 函数库。例如，要连接服务器，采用面向对象的方法，书写格式如下：

```
$link=new mysqli("localhost", "my_user", "my_password", "world");
```

其中 my_user 为用户名，my_password 为密码，world 为要连接的数据库。例如要连接 bookstore 数据库，采用以下语句：

```
$link=new mysqli("localhost", "root", "123456", "bookstore");
```

如果采用面向过程的方法书写，则书写格式如下：

```
$link=mysqli_connect("localhost", "my_user", "my_password", "world");
```

例如，如果要选择一个数据库，使用面向对象的方法如下：

```
$link->select_db("bookstore");
```

使用面向过程的方法如下：

```
mysqli_select_db($link,"bookstore")
```



由于篇幅限制，本书不详细介绍每个 mysqli 函数库的使用，读者可以参考 PHP 帮助手册。

下面通过两个实例介绍如何通过 mysqli 函数库进行基本的操作。

实例 9-25　采用面向过程的方式使用 mysqli 函数库

本实例演示如何通过面向过程的方法来使用 mysqli 函数库，如代码 9-25 所示。

【代码 9-25】　mysqli_proc.php

```
<?php
$link = mysqli_connect('localhost', 'root', '123456');
```

```
//或者使用$link = mysqli_connect('localhost', 'root', '123456',"bookstore")
//使用该语句，将不需要调用 mysqli_select_db 语句

//连接成功后，输出服务器信息
if (mysqli_connect_errno()) {
   die ('服务器连接失败：' . mysql_error());
}
echo '服务器信息：' .mysqli_get_host_info($link) ."<br>";  //输出服务器信息
//设定当前的连接数据库为 bookstore
if (mysqli_select_db($link,'bookstore'))
    echo "已经选择数据库 bookstore<br>";
else
    echo ('数据库选择失败 : ' . mysqli_error());

//插入新记录
$myquery="insert into orders (member_id,item_id,quantity) values
        (1,6,888)";
$result=mysqli_query($link,$myquery) or die("记录插入失败!");

//插入新记录
$myquery="insert into orders (member_id,item_id,quantity) values
        (2,6,888)";
$result=mysqli_query($link,$myquery) or die("记录插入失败!");

//设定查询语句
$myquery="select * from orders";
$result=mysqli_query($link,$myquery);

//获取总记录数量
 $row_cnt = mysqli_num_rows($result);

echo "orders 表总记录数: $row_cnt <br>";

//获取字段数量
$fields = mysqli_num_fields($result);

echo "orders 表的字段数量为：$fields<br>";
//输出表格，用于显示字段名称和记录
echo "<table border=1><tr>";
//循环获取字段名称
while ($finfo = mysqli_fetch_field($result)) {
      echo "<td> $finfo->name </td>";
}
echo "</tr>";
//循环输出记录
//创建记录集
$row=mysqli_fetch_array($result);

//循环输出记录
while ($row)
{
    echo "<tr><td> $row[0] </td> <td> $row[1] </td>";
```

```
    echo "<td> $row[2] </td> <td> $row[3] </td></tr>";
    $row=mysqli_fetch_array($result);
}

mysqli_close($link);  //关闭连接
?>
```

执行结果如图 9-26 所示。



图 9-26　采用面向过程的方式使用 mysqli 函数库

实例 9-26　采用面向对象的方式使用 mysqli 函数库

本实例通过面向对象的方法来使用 mysqli 函数库，如代码 9-26 所示。

【代码 9-26】　mysqli_object.php

```
<?php
$link = new mysqli('localhost', 'root', '123456');
//或者使用$link = mysqli_connect('localhost', 'root', '123456',"bookstore")
//使用该语句，将不需要调用 mysqli_select_db 语句

//连接成功后，输出服务器信息
if (mysqli_connect_errno()) {
   die ('服务器连接失败：' . mysql_error());
}
echo '服务器信息：' .mysqli_get_host_info($link) ."<br>";  //输出服务器信息


//设定当前的连接数据库为 bookstore
if ($link->select_db('bookstore'))
    echo "已经选择数据库 bookstore<br>";
else
    echo ('数据库选择失败 : ' . mysqli_error());

//插入新记录
$myquery="insert into orders (member_id,item_id,quantity) values
        (1,6,777)";
$result=$link->query($myquery) or die("记录插入失败!");
```

```
//插入新记录
$myquery="insert into orders (member_id,item_id,quantity) values
        (2,6,777)";
$result=$link->query($myquery) or die("记录插入失败!");

//设定查询语句
$myquery="select * from orders";
$result=$link->query($myquery);

//获取总记录数量
 $row_cnt = $result->num_rows;
echo "orders 表总记录数: $row_cnt <br>";
//获取字段数量
$fields = $result->field_count;

echo "orders 表的字段数量为：$fields<br>";
//输出表格，用于显示字段名称和记录
echo "<table border=1><tr>";
//循环获取字段名称
while ($finfo = $result->fetch_field()) {
      echo "<td> $finfo->name </td>";
}
echo "</tr>";
//循环输出记录
//创建记录集
$row=$result->fetch_array();

//循环输出记录
while ($row)
{
    echo "<tr><td> $row[0] </td> <td> $row[1] </td>";
    echo "<td> $row[2] </td> <td> $row[3] </td></tr>";
    $row=$result->fetch_array();
}

$link->close();  //关闭连接
?>
```

执行结果如图 9-27 所示。

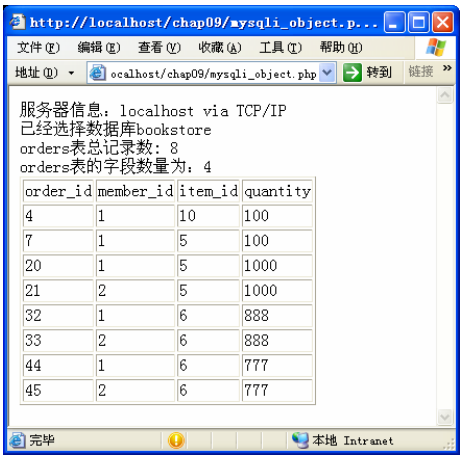


图 9-27　面向对象方法使用 mysqli 函数库

9.3 通过 PHP 连接 SQL Server 数据库

虽然 PHP+MySQL 进行网络开发是最佳选择，但是很可能读者已经熟悉 SQL Server，直接想让 SQL Server 做后台数据库。或者要求必须使用 SQL Server，此时就需要通过 PHP 连接 SQL Server 数据库。

由于篇幅限制，本章将简单介绍 PHP 和 SQL Server 联合开发的内容。

PHP 同 SQL Server 数据库的连接步骤同 MySQL 基本类似，而且函数结构及声明也基本类似，如 MySQL 连接服务器的函数是 mysql_connect，而连接 SQL Server 的函数是 mssql_connect，只要将 mysql 替换成 mssql 即可。

为了能够连接 SQL Server，必须修改 php.ini 文件的设置（因为默认情况下，是不能连接 SQL Server 数据库的），如图 9-28 所示。

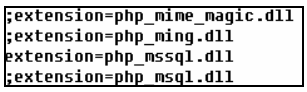


图 9-28 修改 php.ini 的配置

在图 9-28 中，需要将“extension=php_mysqli.dll”语句前面的分号去掉。然后重新启动 Apache 服务器。

连接 SQL Server 的函数为 mssql_connect，该函数声明如下：

```
resource mssql_connect ( [string servername [, string username [, string password]]] )
```

其中参数 servername 表示服务器名称，username 表示用户名，password 表示密码，如要连接本地机器安装的一台 SQL Server 服务器，可以使用如下形式：

```
mssql_connect("localhost","sa","");
```

其中 sa 为 SQL Server 用户名，密码为空。

实例 9-27 连接 SQL Server 数据库

本实例演示了如何在 PHP 中连接 SQL Server 数据库，如代码 9-27 所示。

【代码 9-27】 mssql_connect.php

```
<?php
//其中 amipig 是服务器名称，通常是域名或者机器名
mssql_connect("amipig","sa","") or die("不能连接到 SQL Server");
echo "数据库连接成功!<br>";
?>
```

如果连接成功，则出现如图 9-29 所示结果。



图 9-29 测试是否连接上 SQL Server 数据库

接下来将通过一个比较完整的实例介绍 SQL Server 数据库的应用。为了方便读者学习，

这里将连接 SQL Server 的文件写成了一个类。

实例 9-28　SQL Server 数据库编程

本实例将演示如何在 PHP 中进行 SQL Server 编程，如代码 9-28 所示。

【代码 9-28】　mssql.php

```
<?php
class DB {
    var $Host = "amipig"; // 服务器名称
    var $Database = "pubs"; // 数据库名称
    var $User = "sa"; // 用户名
    var $Password = ""; //密码，读者需要根据实际情况修改以上设置

    var $Link_ID = 0; // 初始连接编号
    var $Query_ID = 0; // 初始查询编号
    var $Row = 0; // 当前记录编号
    var $Errno = 0; // 错误状态
    var $Error = "";
    var $AffNum=0;
    /************************************
     *打印错误方法：显示页面处理的错误信息。
    ************************************/
    function Halt($msg) {
        printf("</td></tr></table><b>Database error:</b> %s<br>n", $msg);
        printf("<b>mssql Error</b>: %s (%s)<br>n",
        $this->Errno,
        $this->Error);
        die("Session halted.");
    }

    /************************************
    *连接数据库，并且选择默认的数据库
    ************************************/
    function Connect(){
        if ( 0 == $this->Link_ID ) {
$this->Link_ID=mssql_connect($this->Host,$this->User,$this->Password) or
                die("Couldn't connect to SQL Server on $servername");
            $db=@mssql_select_db($this->Database,$this->Link_ID);
            if (!$this->Link_ID) {
                $this->Halt("Link-ID == false, mssql_connect failed");
            }
        }
    }

    /************************************
    *关闭数据库，如果数据库连接已经打开则关闭它
    *在调用 Connect()并处理后使用 Close()
    ************************************/
    function Close(){
```

```
            if (0 != $this->Link_ID){
                mssql_close();
            }
        }

        /**********************************************
        *输入 sql 语句，比如 select,update,insert,delete
        *包括存储过程也可以通过这个方法来调用
        **********************************************/
        function Query($Query_String) {
            $this->Connect();

            $this->Query_ID = mssql_query($Query_String);
            $this->Row = 0;
            if (!$this->Query_ID) {
                $msg=mssql_get_last_message();

                if($msg==null || $msg=="")
                {
                    $this->AffNum=1;
                    return 1;
                }

                if(strtolower(substr($Query_String,0,6))!="select"){
                    $this->AffNum=1;
                    return 1;
                }

                $this->Errno = 1;
                $this->Error = "General Error (The mssql interface cannot return
detailed error messages)(".$msg.").";
                $this->halt("Invalid SQL: ".$Query_String);
            }
            return $this->Query_ID;
        }

        /*****************************************************
        *把查询数据库的指针移到下一条记录
        *****************************************************/
        function NextRecord(){
            $this->Record = array();
            mssql_next_result($this->Query_ID);
            $this->Record=mssql_fetch_array($this->Query_ID);

            $result=$this->Record;
            if(!is_array($result)) return $this->Record;
            foreach($result as $key => $value){
                $keylower=strtolower($key);
                if($keylower!=$key) $this->Record[$keylower]=$value;
            }
```

```
        return $this->Record;
    }

    /*********************************************************
    *重新定位查询数据库的指针
    *********************************************************/
    function Seek($pos) {
        if($pos<=0) return;
        if(eregi("[0-9]",$pos)) mssql_data_seek($this->Query_ID,$pos);
    }

    /*********************************************************
    *获取查询数据库得到的总行数
    *********************************************************/
    function NumRows(){
    if($this->Query_ID) $num_rows=mssql_num_rows($this->Query_ID);
        else $num_rows=$this->AffNum;

        return $num_rows;
    }

    /*********************************************************
    *字段数
    *********************************************************/
    function NumFields(){
        return count($this->Record)/2;
    }

    /*******************************
    *该字段的值
    *******************************/
    function FieldValue($Field_Name){
        return $this->Record[$Field_Name];
    }

    /*******************************
    *update,insert,delete 影响的行数
    *******************************/
     function AffectedRows(){
        if($this->Query_ID) return mssql_num_rows($this->Query_ID);
        else{
            return $this->AffNum;
        }
    }
 }
 ?>
<?php
    //实例应用
    //构造新的 DB 类
```

```
        $DBConn=new DB;

        //写入 sql 查询语句，pubs 数据库是 sql server 自带数据库
        //authors 是 pubs 数据库中的一个表
        $SqlStr="select au_id,au_fname,phone from authors where au_fname like
'%d%'";
        $DBConn->Query($SqlStr);

        //循环输出查询得到的结果
        echo "<table border=1><tr>";
        echo "<td>au_id</td><td>au_fname</td><td>phone</td></tr>";
        while($Row=$DBConn->NextRecord()){
            echo "</tr><td> $Row[0] </td><td> $Row[1] </td>";
            echo "<td> $Row[2] </td></tr>";
        }
       echo "</table>";
      //关闭数据库连接
      $DBConn->Close();
   ?>
```

执行结果如图 9-30 所示。

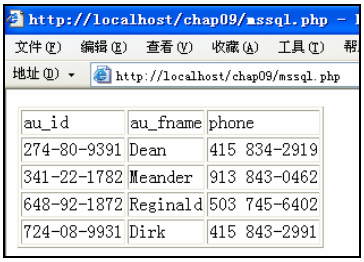


图 9-30　通过 PHP 对 SQL Server 进行操作

以上代码首先创建了一个 DB 类，该类能够进行常用的各种数据库操作。然后应用 DB 类从数据库 pubs 的 authors 表中查询一部分记录并显示出来。

第 10 章　PHP 中的图像处理技术

PHP 不仅能输出 HTML 的文件，还可以创建及操作多种不同图像格式的图像文件，包括 gif、png、jpg、wbmp 和 xpm。更方便的是，PHP 可以直接将图像流输出到浏览器。要处理图像，需要在编译 PHP 时加上图像函数库 GD2。

在 PHP 中有些图像函数是可以直接使用的，但是大多数的函数调用需要安装 GD2 函数库。有关 GD2 函数库的详细信息，可以访问站点 http://www.boutell.com/gd/。从 PHP 的 4.3 版本开始，PHP 就捆绑了自带版本的 GD2。在 Windows 平台下，PNG 和 JPEG 是自动支持的。如果使用 UNIX 同时又希望使用 PNG，必须安装 libpng 库和 zlip 库。

在本书中，将介绍 PHP 捆绑的 GD2 函数库。要使用该函数库，要修改 php.ini 文件，将“extension=php_gd2.dll”选项前面的分号去掉，如图 10-1 所示。

修改以后，重新启动 Apache 服务器。

实例 10-1　测试 GD2 函数库是否安装成功

本实例用于测试是否已经安装了 GD2 函数库，如代码 10-1 所示。

【代码 10-1】　testgd.php

```
<?php
// 建立一幅 100×30 的图像
$im = imagecreatetruecolor(100, 30);
// 设置背景颜色
$bg = imagecolorallocate($im, 0, 0, 0);
//设置字体颜色
$textcolor = imagecolorallocate($im, 0, 255, 255);
// 把字符串写在图像左上角
imagestring($im, 5, 0, 0, "Hello world!", $textcolor);
// 输出图像
header("Content-type: image/jpeg");
imagejpeg($im);
?>
```

testgd.php 的执行结果如图 10-2 所示。

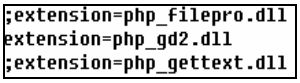


图 10-1　修改 php.ini 的配置



图 10-2　testgd.php 执行结果

10.1 GD2 函数库中的常用函数

下面详细介绍 GD2 函数库中的重要函数。

（1）GetImageSize()函数。该函数取得图片的长宽 ，声明如下：

```
array getimagesize(string filename, array [imageinfo]);
```

getimagesize 函数可用来取得 WWW 上的 GIF、JPEG 及 PNG 3 种图片的高与宽，不需要安装 GD library 就可以使用本函数。传回的阵列有 4 个元素，第 1 个元素（索引值 0）是图片的高度，单位是像素。第 2 个元素（索引值 1）是图片的宽度。第 3 个元素（索引值 2）是图片的档案格式，1 为 GIF 格式、2 为 JPEG/JPG 格式、3 为 PNG 格式。第 4 个元素（索引值 3）为图片的高与宽字串（height=xxx width=yyy）。可省略的参数 imageinfo 用来取得一些图片的相关信息，例如 IPTC（http://www.xe.net/iptc）的 APP13 标记，就可以加在图片中，利用 iptcparse()来解析。

（2）ImageArc()函数。该函数用于画弧线，声明如下：

```
bool imagearc ( resource image, int cx, int cy, int w, int h, int s, int e, int color )
```

imagearc()函数以 cx，cy（图像左上角为 0，0）为中心在 image 所代表的图像中画一个椭圆弧。w 和 h 分别指定了椭圆的宽度和高度，起始和结束点以 s 和 e 参数以角度指定。0° 位于 3 点钟位置，以顺时针方向绘画。

（3）ImageChar()函数。该函数写出横向字符，声明如下：

```
bool imagechar ( resource image, int font, int x, int y, string c, int color )
```

imagechar()函数将字符串 c 的第一个字符画在 image 指定的图像中，其左上角位于 x，y（图像左上角为 0，0），颜色为 color。如果 font 是 1，2，3，4 或 5，则使用内置的字体（更大的数字对应于更大的字体）。

（4）ImageCharUp()函数。该函数写出竖式字符，声明如下：

```
bool imagecharup ( resource image, int font, int x, int y, string c, int color )
```

imagecharup()将字符 c 垂直地画在 image 指定的图像上，位于 x，y（图像左上角为 0，0），颜色为 color。如果 font 为 1，2，3，4 或 5，则使用内置的字体。

（5）ImageColorAllocate()函数。该函数用于匹配颜色，声明如下：

```
int imagecolorallocate ( resource image, int red, int green, int blue )
```

imagecolorallocate()返回一个标识符，代表了由给定的 RGB 成分组成的颜色。image 参数是 imagecreatetruecolor()函数的返回值。red，green 和 blue 分别是所需要的颜色的红，绿，蓝成分。这些参数是 0～255 的整数或者十六进制的 0x00～0xFF。imagecolorallocate()必须被调用，以创建每一种用在 image 所代表的图像中的颜色。

（6）ImageColorTransparent()函数。该函数指定透明背景色，声明如下：

```
int imagecolortransparent ( resource image [, int color] )
```

imagecolortransparent()函数将 image 图像中的透明色设定为 color。image 是 imagecreatetruecolor()返回的图像标识符，color 是 imagecolorallocate()返回的颜色标识符。

（7）ImageCopyResized()函数。该函数复制新图像并调整大小，声明如下：

```
bool imagecopyresized ( resource dst_image, resource src_image, int dst_x, int dst_y, int src_x, int src_y, int dst_w, int dst_h, int src_w, int src_h )
```

imagecopyresized()将一幅图像中的一块正方形区域拷贝到另一个图像中。dst_image 和

src_image 分别是目标图像和源图像的标识符。如果源和目标的宽度和高度不同，则会进行相应的图像收缩和拉伸。坐标指的是左上角。本函数可用来在同一幅图内部拷贝（如果 dst_image 和 src_image 相同），但如果区域交迭的话则结果不可预知。

（8）ImageCreate()函数。该函数建立新图，声明如下：

```
resource imagecreate ( int x_size, int y_size )
```

imagecreate()返回一个图像标识符，代表了一幅大小为 x_size 和 y_size 的空白图像。推荐使用 imagecreatetruecolor()。

（9）ImageDashedLine()函数。该函数用于绘虚线，声明如下：

```
bool imagedashedline ( resource image, int x1, int y1, int x2, int y2, int color )
```

不建议使用本函数。应该用 imagesetstyle()和 imageline()的组合替代。

（10）ImageDestroy()函数。该函数用于销毁一个图形，声明如下：

```
bool imagedestroy ( resource image )
```

imagedestroy()释放与 image 关联的内存。image 是由图像创建函数返回的图像标识符，例如 imagecreatetruecolor()。

（11）ImageFill()函数。该函数用于颜色填充，声明如下：

```
bool imagefill ( resource image, int x, int y, int color )
```

imagefill()在 image 图像的坐标 x，y（图像左上角为 0，0）处用 color 颜色执行区域填充（即与 x，y 点颜色相同且相邻的点都会被填充）。

（12）ImageFilledPolygon()函数。该函数用于画一个多边形并着色，声明如下：

```
bool imagefilledpolygon ( resource image, array points, int num_points, int color )
```

imagefilledpolygon()在 image 图像中画一个填充了的多边形。points 参数是一个按顺序包含多边形各顶点的 x 和 y 坐标的数组。num_points 参数是顶点的总数，必须大于 3。

（13）ImageFilledRectangle()函数。该函数用于画一个矩形并着色，声明如下：

```
bool imagefilledrectangle ( resource image, int x1, int y1, int x2, int y2, int color )
```

imagefilledrectangle()在 image 图像中画一个用 color 颜色填充的矩形，其左上角坐标为（x1，y1），右下角坐标为（x2，y2）。（0，0）是图像的最左上角。

（14）ImageFillToBorder()函数。本函数用于指定颜色区域内着色，声明如下：

```
bool imagefilltoborder ( resource image, int x, int y, int border, int color )
```

imagefilltoborder()从（x，y）（图像左上角为 0，0）点开始用 color 颜色执行区域填充，直到遇到颜色为 border 的边界为止。边界内的所有颜色都会被填充。如果指定的边界色和该点颜色相同，则没有填充。如果图像中没有该边界色，则整幅图像都会被填充。

（15）ImageFontHeight()函数。该函数用于取得字符的高度，声明如下：

```
int imagefontheight ( int font )
```

返回指定字体一个字符高度的像素值。

（16）ImageFontWidth()函数。该函数用于取得字符的宽度，声明如下：

```
int imagefontwidth ( int font )
```

返回指定字体一个字符宽度的像素值。

（17）ImageInterlace()函数。该函数用于激活或禁止隔行扫描，声明如下：

```
int imageinterlace ( resource image [, int interlace] )
```

imageinterlace()打开或关闭隔行扫描的比特标记。如果 interlace 为 1 则图像为隔行扫描的，

如果 interlace 为 0 则图像为非隔行扫描的。

如果设定了隔行扫描比特标记而图像使用 JPEG 格式，则图像被创建为渐进式 JPEG。本函数返回图像中是否设定了隔行扫描比特标记。

（18）ImageLine()函数。该函数用于绘实线，声明如下：

```
bool imageline ( resource image, int x1, int y1, int x2, int y2, int color )
```

imageline()用 color 颜色在图像 image 中从坐标（x1，y1）到（x2，y2）（图像左上角为 0，0）画一条线段。

（19）ImageLoadFont()函数。该函数用于载入点阵字型，声明如下：

```
int imageloadfont ( string file )
```

imageloadfont()加载一个用户定义的位图字体并返回该字体的标识符（其值总是大于 5，因此不会和内置字体冲突）。在产生错误的情况下，该函数返回 FALSE。

（20）ImagePolygon()函数。该函数用于绘制多边形，声明如下：

```
bool imagepolygon ( resource image, array points, int num_points, int color )
```

imagepolygon()在图像中创建一个多边形。points 是一个 PHP 数组，包含了多边形的各个顶点坐标，即 points[0] = x0，points[1] = y0，points[2] = x1，points[3] = y1，依此类推。num_points 是顶点的总数。

（21）ImageRectangle()函数。该函数用于绘制矩形，声明如下：

```
bool imagerectangle ( resource image, int x1, int y1, int x2, int y2, int col )
```

imagerectangle()用 col 颜色在 image 图像中画一个矩形，其左上角坐标为（x1，y1），右下角坐标为（x2，y2）。图像的左上角坐标为（0，0）。

（22）ImageSetPixel()函数。该函数用于绘制点，声明如下：

```
bool imagesetpixel ( resource image, int x, int y, int color )
```

imagesetpixel()在 image 图像中用 color 颜色在（x，y）坐标（图像左上角为 0，0）上画一个点。

（23）ImageString()函数。该函数用于水平地画一行字符串，声明如下：

```
bool imagestring ( resource image, int font, int x, int y, string s, int col )
```

imagestring()用 col 颜色将字符串 s 画到 image 所代表的图像的（x，y）坐标处（这是字符串左上角坐标，整幅图像的左上角为 0，0）。如果 font 是 1，2，3，4 或 5，则使用内置字体。

（24）ImageStringUp()函数。该函数用于垂直地画一行字符串，声明如下：

```
bool imagestringup ( resource image, int font, int x, int y, string s, int
col )
```

imagestring()用 col 颜色将字符串 s 垂直地画到 image 所代表的图像的（x，y）坐标处（图像的左上角为 0，0）。如果 font 是 1，2，3，4 或 5，则使用内置字体。

（25）ImageSX()函数。该函数用于取得图片的宽度，声明如下：

```
int imagesx ( resource image )
```

imagesx()返回 image 所代表的图像的宽度。

（26）ImageSY()函数。该函数取得图片的高度，声明如下：

```
int imagesy ( resource image )
```

参数 im 表示图片的 handle。

（27）ImageTTFBBox()函数。该函数计算 TTF 文字所占区域，声明如下：

```
array imagettfbbox ( float size, float angle, string fontfile, string text )
```

本函数计算并返回一个包围着 TrueType 文本范围的虚拟方框的像素大小。size，像素单位的字体大小。angle，text 将被度量的角度大小。fontfile，TrueType 字体文件的文件名（可以是 URL）。text，要度量的字符串。

（28）ImageTTFText()函数。该函数用于将 TTF 文字写到图中，声明如下：

```
    array imagettftext ( resource image, float size, float angle, int x, int y,
int color, string fontfile, string text )
```

imagettftext()返回一个含有 8 个单元的数组，表示文本外框的 4 个角，顺序为左下角，右下角，右上角，左上角。这些点是相对于文本的和角度无关，因此“左上角”指的是以水平方向看文字时其左上角。

image，图像资源。参见 imagecreatetruecolor()函数。size，字体大小。根据 GD 版本不同，应该以像素大小指定（GD1）或点大小（GD2）。angle，角度制表示的角度，0 度为从左向右读的文本。更高数值表示逆时针旋转。例如 90 度表示从下向上读的文本。x，由（x，y）所表示的坐标定义了第一个字符的基本点（大概是字符的左下角）。这和 imagestring()不同，其（x，y）定义了第一个字符的左上角。例如"top left"为 0，0。y，y 坐标。它设定了字体基线的位置，不是字符的最底端。color，颜色索引。使用负的颜色索引值具有关闭防锯齿的效果。参见 imagecolorallocate()函数。fontfile，想要使用的 TrueType 字体的路径。text，文本字符串。

（28）ImageColorAt()函数。该函数取得图中指定点颜色的索引值，声明如下：

```
    int imagecolorat ( resource image, int x, int y )
```

返回 image 所指定的图形中指定位置像素的颜色索引值。如果 PHP 编译时加上了 GD 库 2.0 或更高的版本并且图像是真彩色，则本函数以整数返回该点的 RGB 值。用移位加掩码来取得红、绿、蓝各自成分的值。

（29）ImageColorClosest()函数。该函数计算色表中与指定颜色最接近的，声明如下：

```
    int imagecolorclosest ( resource image, int red, int green, int blue )
```

返回图像调色板中与指定的 RGB 值最“接近”的颜色。指定的颜色与调色板中的每个颜色的“距离”的计算方法是把 RGB 值当成三维空间中点的坐标。

如果从文件创建了图像，只有图像中使用了的颜色会被辨别，仅出现在调色板中的颜色不会被辨别。

（30）ImageColorExact()函数。该函数计算色表上指定颜色索引值，声明如下：

```
    int imagecolorexact ( resource image, int red, int green, int blue )
```

返回图像调色板中指定颜色的索引值。如果颜色不在图像的调色板中，返回–1。

如果从文件创建了图像，只有图像中使用了的颜色会被辨别，仅出现在调色板中的颜色不会被辨别。

（31）ImageColorResolve()函数。该函数用于计算色表上指定或最接近颜色的索引值，声明如下：

```
    int imagecolorresolve ( resource image, int red, int green, int blue )
```

本函数可以保证对所请求的颜色返回一个颜色索引，要么是确切值要么是所能得到最接近的替代值。

（32）ImageColorSet()函数。该函数用于设定色表上指定索引的颜色，声明如下：

```
    void imagecolorset ( resource image, int index, int red, int green, int blue )
```

函数将调色板中指定的索引设定为指定的颜色。对于在调色板图像中创建类似区域填充（flood-fill）的效果很有用，免去了真正进行填充的开销。

（33）ImageColorsForIndex()函数。该函数用于取得色表上指定索引的颜色，声明如下：

```
array imagecolorsforindex ( resource image, int index )
```

函数返回一个具有 red，green，blue 和 alpha 键名的关联数组，包含了指定颜色索引的相应的值。

（34）ImageColorsTotal()函数。该函数用于计算图像的颜色数，声明如下；

```
int imagecolorstotal ( resource image )
```

函数返回指定图像的调色板中的颜色数目。

（35）ImagePSLoadFont()函数。该函数载入 PostScript 字型，声明如下：

```
resource imagepsloadfont ( string filename )
```

如果一切正常，将返回一个合法的字体索引以备使用；否则返回 FALSE 并显示一条信息说明哪里错误，但不能直接读取此信息，因为输出格式是图像。

（36）ImagePSFreeFont()函数。该函数用于卸载 PostScript 字型，声明如下：

```
bool imagepsfreefont ( resource fontindex )
```

imagepsfreefont()释放一个 PostScript Type 字体所占用的内存。

（37）ImagePSEncodeFont()函数。该函数将 PostScript 字型转成向量字，声明如下：

```
bool imagepsencodefont ( resource font_index, string encodingfile )
```

从文件中加载字符编码矢量并用其替换给定字体的编码矢量。

（38）ImagePSText()函数。该函数用于将 PostScript 文字写到图中，声明如下：

```
array imagepstext ( resource image, string text, resource font, int size, int
foreground, int background, int x, int y [, int space, int tightness, float angle,
int antialias_steps] )
```

foreground 是文本的颜色，background 是文本以防锯齿（antialiasing）方式淡入的颜色。以 background 为颜色的像素实际上不会被画上去，所以背景图像不需要是实心的颜色。

以（x，y）给出的坐标定义了第一个字符的起点（或参考点，基本是字符的左下角）。这和 imagestring()不同，其（x，y）坐标定义的是第一个字符的右上角。

（39）ImagePSBBox()函数。该函数计算 PostScript 文字所占区域，声明如下：

```
array imagepsbbox ( string text, int font, int size [, int space, int tightness,
float angle] )
```

size 以像素表示。space 可以用来改变字体中默认间距的值。此值将被加到通常的值上，可以为负值。tightness 可以控制字符之间的间距。此值将被加到通常字符宽度上，可以为负值。angle 以角度表示。space 和 tightness 以字符间距单元表示。space，tightness 和 angle 参数为可选项。

（40）ImageCreateFromPNG()函数。该函数取出 PNG 图形，声明如下：

```
resource imagecreatefrompng ( string filename )
```

函数返回图像标识符，代表从给定的文件名取得的图像，失败时返回一个空字符串，并且输出一条错误信息。

（41）ImagePNG()函数。该函数用于建立 PNG 图形，声明如下：

```
bool imagepng ( resource image [, string filename] )
```

函数将 GD 图像流(image)以 PNG 格式输出到标准输出(通常为浏览器)，如果用 filename 给出了文件名则将其输出到该文件。

（42）ImageCreateFromGIF()函数。该函数用于取出 GIF 图形，声明如下：

```
resource imagecreatefromgif ( string filename )
```

函数返回图像标识符，代表了从给定的文件名取得的图像，失败时返回一个空字符串，

并且输出一条错误信息。

（43）ImageGIF()函数。该函数用于建立 GIF 图形，声明如下：

```
bool imagegif ( resource image [, string filename] )
```

函数从 image 图像以 filename 为文件名创建一个 GIF 图像。image 参数是 imagecreate-truecolor()函数的返回值。图像格式为 GIF87a。如果使用了 imagecolortransparent()使图像为透明，则其格式为 GIF89a。

filename 参数为可选，如果省略，则原始图像流将被直接输出。通过 header()发送“Content-type: image/gif”可以使 PHP 脚本直接输出 GIF 图像。

以上介绍了 GD2 函数库中大部分函数声明及含义，接下来介绍如何使用这些函数。

10.2　利用 GD2 函数库进行图像处理

10.2.1　创建图像

应用 GD2 进行图形处理的第 1 步就是要创建一个图像，通常创建一个图像需要以下几个步骤。

（1）创建一个背景图像，之后所有的操作都基于该背景。

（2）在背景上绘制图形轮廓或者输入文本。

（3）输出最终图形。

（4）清除所有资源。

实例 10-2　创建一幅图像

本实例演示了创建一幅图像的基本步骤，如代码 10-2 所示。

【代码 10-2】　createimage.php

```
<?php
// 建立多边形各顶点坐标的数组
$values = array(
          40,  50,  // Point 1 (x, y)
          20,  240, // Point 2 (x, y)
          60,  60,  // Point 3 (x, y)
          240, 20,  // Point 4 (x, y)
          50,  40,  // Point 5 (x, y)
          10,  10   // Point 6 (x, y)
          );

// 创建图像
$image = imagecreatetruecolor(250, 250);

// 设定颜色
$bg   = imagecolorallocate($image, 200, 200, 200);
$blue = imagecolorallocate($image, 0, 0, 255);

// 画一个多边形
imagefilledpolygon($image, $values, 6, $blue);
```

```
// 输出图像
header('Content-type: image/png');
imagepng($image);
imagedestroy($image);
?>
```

执行结果如图 10-3 所示。

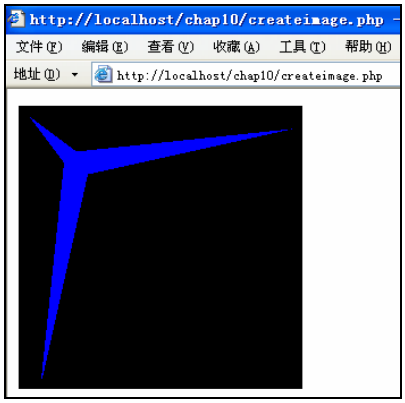


图 10-3　创建图像并显示

1．创建一幅背景图像

创建背景图像通常有两种方法，一种是调用 imagecreatetruecolor 函数创建一个空白图像，例如在 createimage.php 中的“$image = imagecreatetruecolor(250, 250);”语句。

第 2 种方法就是读入一个已有的图像文件，然后对图像进行过滤、改变大小或者添加其他图像等。根据读入数据的图像格式不同，可以调用下面 3 个函数来实现，imagecreatefrompng、imagecreatefromjpeg 或者 imagecreatefromgif。

imagecreatefrompng 函数从 GIF 文件或 URL 新建一幅图像，声明如下：

```
resource imagecreatefromgif (string filename)
```

imagecreatefromjpeg 从 JPEG 文件或 URL 新建一幅图像，声明如下：

```
resource imagecreatefromjpeg (string filename )
```

imagecreatefromgif 从 GIF 文件或 URL 新建一幅图像，声明如下：

```
resource imagecreatefromgif ( string filename )
```

2．在图像上绘图或者打印文字

在创建图像背景后，可以在背景上面进行各种操作，例如绘制图形或者打印文字。在进行操作之前需要设定需要绘制的颜色，设定颜色使用 imagecolorallocate 函数，例如 createimage.php 文件中的颜色设定语句：

```
// 设定颜色
$bg   = imagecolorallocate($image, 200, 200, 200);
$blue = imagecolorallocate($image, 0, 0, 255);
```

设定完颜色之后，便可以进行各种操作，如绘制图形。

createimage.php 文件中的图形绘制语句如下：

```
// 画一个多边形
imagefilledpolygon($image, $values, 6, $blue);
```

3．输出最终图像

创建完图像后，需要显示给客户端，通常的做法是保存到一个文件或者输出到浏览器。

如果是输出到浏览器，必须在发送到浏览器的时候，通知浏览器当前发送的类型，可以通过 Header()函数来指定图像的 MIME 类型，如下所示：

```
header('Content-type: image/png');
```

对于普通的 HTML 文件，服务器最先发送的是如下语句：

```
header('Content-type: text/html');
```

在 PHP 中，Header()函数用来发送页面的 HTTP 标题。



注意　在调用 Header()函数的时候，如果该页的 HTTP 标题已经被发送，那么该函数则不会被执行。

在发送数据类型之后，需要将图像显示在浏览器中，如下所示：

```
imagepng($image);
```

ImagePNG 函数以 PNG 格式将输出内容发送到浏览器。如果希望以不同的格式发送，还可以调用其他函数，如 ImageJPEG()函数，但是此时需要修改 Header()函数的内容，如下所示：

```
header('Content-type: image/jpeg');
```

如果是将图像保存到一个文件中，可以使用如下语句：

```
imagepng($image,filename);
```

4．释放资源

在完成图像处理之后，需要释放图像资源，释放资源将调用 imagedestroy 函数，如下所示：

```
imagedestroy($image);
```

10.2.2　图像处理的各种实例

前面介绍了创建图像的基本步骤，接下来将介绍各种常用的图像处理技术。

普通页面在显示的时候，服务器已经发送了一个 Header，如果此时再发送一个显示图像的 Header，则浏览器是不做处理的。因此显示图片的一种方法就是将其保存成一幅临时图像，然后显示出来。

实例 10-3　将图像保存到文件并显示

本实例演示了如何将创建好的一幅图像保存到文件，并通过浏览器显示该文件，如代码 10-3 所示。

【代码 10-3】　showpic_1.php

```
<?php
// 创建一个 200×200 的图像
$img = imagecreatetruecolor(200, 200);
// 分配颜色
$white = imagecolorallocate($img, 255, 255, 255);
$black = imagecolorallocate($img, 200, 200, 100);
// 画一个圆
imagearc($img, 100, 100, 150, 150, 0, 360, $black);
// 将图像保存到文件
imagepng($img,"exam02.png");
// 释放内存
```

```
imagedestroy($img);
//调用 HTML 的<image>标记显示图像
?>
<img src="exam02.png">
```

执行结果如图 10-4 所示。

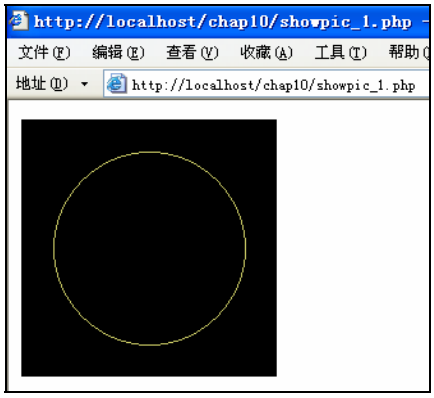


图 10-4 显示图像文件

实例 10-4 直接在标记中包含 PHP 文件

本实例由两个文件组成，一个文件用来创建图像，但是并不输出，文件名为 showpic_2.php，如代码 10-4 所示。另一个文件名为 showpic_3.htm，用来调用 showpic_2.php 输出图像，如代码 10-5 所示。

【代码 10-4】 showpic_2.php

```
<?php
// 创建一个 200×200 的图像
$img = imagecreatetruecolor(200, 200);
// 分配颜色
$white = imagecolorallocate($img, 255, 255, 255);
$black = imagecolorallocate($img, 200, 200, 100);
// 画一个圆
imagearc($img, 100, 100, 150, 150, 0, 360, $black);
// 将图像输出到浏览器
header("Content-type: image/png");
imagepng($img);
// 释放内存
imagedestroy($img);
//调用 HTML 的<image>标记显示图像
?>
```

【代码 10-5】 showpic_3.htm

```
调用 showpic_2.php 显示图像<br>
<img src="showpic_2.php">
```

执行结果如图 10-5 所示。

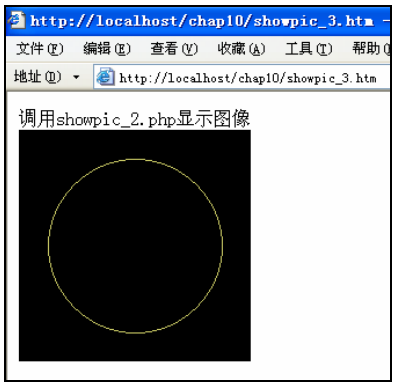


图 10-5　调用 showpic_2.php 显示图像

实例 10-5　为图像着色

本实例演示如何为图像的不同区域着色，如代码 10-6 所示。

【代码 10-6】　deploycolor.php

```
<?php
$size = 300;
$image=imagecreatetruecolor($size, $size);
// 用白色背景加黑色边框画方框
$back = imagecolorallocate($image, 255, 255,255);
$border = imagecolorallocate($image, 0, 0, 0);
imagefilledrectangle($image, 0, 0, $size - 1,$size - 1, $back);
imagerectangle($image, 0, 0, $size - 1, $size -1, $border);
$yellow_x = 100;
$yellow_y = 75;
$red_x    = 120;
$red_y    = 165;
$blue_x   = 187;
$blue_y   = 125;
$radius   = 150;
// 用 alpha 值分配一些颜色
$yellow = imagecolorallocatealpha($image, 255, 255, 0, 75);
$red    = imagecolorallocatealpha($image, 255, 0, 0, 75);
$blue   = imagecolorallocatealpha($image, 0, 0, 255, 75);
// 画 3 个交叠的圆
imagefilledellipse($image, $yellow_x, $yellow_y, $radius, $radius, $yellow);
imagefilledellipse($image, $red_x, $red_y,$radius, $radius, $red);
imagefilledellipse($image, $blue_x, $blue_y,$radius, $radius, $blue);
// 不要忘记输出正确的 header
header('Content-type: image/png');
// 最后输出结果
imagepng($image);
imagedestroy($image);
?>
```

执行结果如图 10-6 所示。

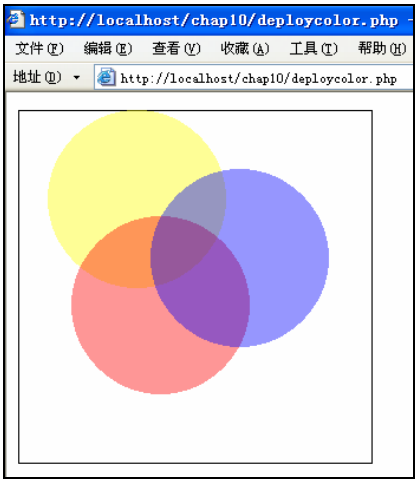


图 10-6 为图像着色

实例 10-6 获取像素颜色

本实例演示如何获取指定像素的颜色，如代码 10-7 所示。

【代码 10-7】 getcolor.php

```
<?php
$im = ImageCreateFromPng("exam01.png");
$rgb = ImageColorAt($im, 100, 100);
$r = ($rgb >> 16) & 0xFF;
$g = ($rgb >> 8) & 0xFF;
$b = $rgb & 0xFF;
echo "红： $r 绿: $g 蓝： $b";
?>
```

执行结果如图 10-7 所示。



图 10-7 获取像素颜色

实例 10-7 根据字符串流创建图像

本实例演示如何从字符串中创建一幅图像，如代码 10-8 所示。

【代码 10-8】 imagecreatefromstring.php

```
<?php
$data = 'iVBORw0KGgoAAAANSUhEUgAAABwAAAASCAMAAAB/2U7WAAAABl'
      . 'BMVEUAAAD///+l2Z/dAAAASUlEQVR4XqWQUQoAIAxC2/0vXZDr'
      . 'EX4IJTRkb7lobNUStXsB0jIXIAMSsQnWlsV+wULF4Avk9fLq2r'
      . '8a5HSE35Q3eO2XP1A1wQkZSgETvDtKdQAAAABJRU5ErkJggg==';
```

```
$data = base64_decode($data);

$im = imagecreatefromstring($data);
if ($im !== false) {
    header('Content-Type: image/png');
    imagepng($im);
}
else {
    echo 'An error occured.';
}
?>
```

执行结果如图 10-8 所示。



图 10-8　从字符流创建图像

实例 10-8　画一个 3D 效果的饼图

本实例演示如何制作一个三维效果的饼图，如代码 10-9 所示。

【代码 10-9】　imagefilledarc.php

```
<?php
// 创建图像
$image = imagecreatetruecolor(100, 100);

// allocate some solors
$white    = imagecolorallocate($image, 0xFF, 0xFF, 0xFF);
$gray     = imagecolorallocate($image, 0xC0, 0xC0, 0xC0);
$darkgray = imagecolorallocate($image, 0x90, 0x90, 0x90);
$navy     = imagecolorallocate($image, 0x00, 0x00, 0x80);
$darknavy = imagecolorallocate($image, 0x00, 0x00, 0x50);
$red      = imagecolorallocate($image, 0xFF, 0x00, 0x00);
$darkred  = imagecolorallocate($image, 0x90, 0x00, 0x00);

//制作 3D 效果
for ($i = 60; $i > 50; $i--) {
  imagefilledarc($image, 50, $i, 100, 50, 0, 45, $darknavy, IMG_ARC_PIE);
  imagefilledarc($image, 50, $i, 100, 50, 45, 75 , $darkgray, IMG_ARC_PIE);
  imagefilledarc($image, 50, $i, 100, 50, 75, 360 , $darkred, IMG_ARC_PIE);
}

imagefilledarc($image, 50, 50, 100, 50, 0, 45, $navy, IMG_ARC_PIE);
imagefilledarc($image, 50, 50, 100, 50, 45, 75 , $gray, IMG_ARC_PIE);
imagefilledarc($image, 50, 50, 100, 50, 75, 360 , $red, IMG_ARC_PIE);
// flush image
header('Content-type: image/png');
imagepng($image);
```

```
imagedestroy($image);
?>
```

执行结果如图 10-9 所示。



图 10-9　创建一个三维饼图

实例 10-9　制作定义按钮

本实例将介绍如何创建一个自定义图像按钮。本实例分成两个文件，一个是 HTM 文件，用于让用户选择按钮类型，文件名为 design_button.html，如代码 10-10 所示。另外一个用于输出按钮结果，文件名为 make_button.php，如代码 10-11 所示。

【代码 10-10】　design_button.html

```
<html>
<head>
 <title>Create buttons</title>
<head>
<body>
<h1>创建按钮</h1>
<form method="post" action="make_button.php">
输入按钮文字:<br />
<input type="text" name="button_text"><br /><br />
选择按钮颜色:<br />
<input type="radio" name="color" value="red">红<br />
<input type="radio" name="color" value="green">绿<br />
<input type="radio" name="color" value="blue">蓝<br /><br />
<input type="submit" value="创建按钮">
</form>
</body>
```

执行结果如图 10-10 所示。



图 10-10　design_button.html 文件执行结果

单击“创建按钮”后，可提交到 make_button.php 文件进行处理。

【代码 10-11】　make_button.php

```
<?php
$button_text = $_REQUEST['button_text'];  //获取按钮文本
$color = $_REQUEST['color'];  //获取设定颜色

if (empty($button_text) || empty($color))
{
  echo 'Could not create image - form not filled out correctly';
  exit;
}

// 从图像创建一个背景
$im = ImageCreateFromPNG ($color.'-button.png');

$width_image = ImageSX($im);
$height_image = ImageSY($im);

// 设定图像大小
$width_image_wo_margins = $width_image - (2 * 18);
$height_image_wo_margins = $height_image - (2 * 18);

// 设定字体大小
$font_size = 33;

//设定字体路径
putenv('GDFONTPATH=C:\WINDOWS\Fonts');
$fontname = 'arial';

do
{
  $font_size--;

  // 获取字体大小
  $bbox=ImageTTFBBox ($font_size, 0, $fontname, $button_text);

  $right_text = $bbox[2];   // right co-ordinate
  $left_text = $bbox[0];    // left co-ordinate
  $width_text = $right_text - $left_text;  // 宽度
  $height_text = abs($bbox[7] - $bbox[1]);  // 高度

}
while ( $font_size>8 &&
      ( $height_text>$height_image_wo_margins ||
        $width_text>$width_image_wo_margins )
     );

if ( $height_text>$height_image_wo_margins ||
    $width_text>$width_image_wo_margins )
{
```

```
  // no readable font size will fit on button
  echo 'Text given will not fit on button.<br />';
}
else
{
  $text_x = $width_image/2.0 - $width_text/2.0;
  $text_y = $height_image/2.0 - $height_text/2.0 ;

  if ($left_text < 0)
     $text_x += abs($left_text);

  $above_line_text = abs($bbox[7]);
  $text_y += $above_line_text;

  $text_y -= 2;

  $white = ImageColorAllocate ($im, 255, 255, 255);

  ImageTTFText ($im, $font_size, 0, $text_x, $text_y, $white, $fontname,
             $button_text);

  Header ('Content-type: image/png');
  ImagePNG ($im);
}

ImageDestroy ($im);
?>
```

执行结果如图 10-11 所示。



图 10-11　创建自定义按钮

实例 10-10　设置线型

本实例用于各种线型的设置，如代码 10-12 所示。

【代码 10-12】　line.php

```
<?php
header("Content-type: image/png");
$im  = imagecreatetruecolor(100, 100);
$w   = imagecolorallocate($im, 255, 255, 255);
$red = imagecolorallocate($im, 255, 0, 0);
/* 画一条虚线，5 个红色像素，5 个白色像素 */
$style = array($red, $red, $red, $red, $red, $w, $w, $w, $w, $w);
imagesetstyle($im, $style);
imageline($im, 0, 0, 100, 100, IMG_COLOR_STYLED);
```

```
    /* 用 imagesetbrush()和 imagesetstyle 画一行笑脸 */
    $style = array($w, $w, $w, $w, $w, $w, $w, $w, $w, $w, $w, $w, $red);
    imagesetstyle($im, $style);
    $brush = imagecreatefrompng("http://www.libpng.org/pub/png/images/smile.
happy.png");
    $w2 = imagecolorallocate($brush, 255, 255, 255);
    imagecolortransparent($brush, $w2);
    imagesetbrush($im, $brush);
    imageline($im, 100, 0, 0, 100, IMG_COLOR_STYLEDBRUSHED);

    imagepng($im);
    imagedestroy($im);
    ?>
```

执行结果如图 10-12 所示。

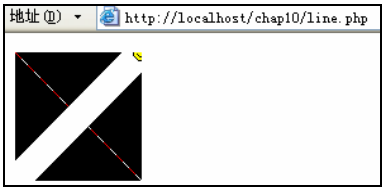


图 10-12　line.php 执行结果

实例 10-11　显示一个温度计

本实例介绍如何显示一个漂亮的温度计，如代码 10-13 所示。

【代码 10-13】　thermometer.php

```
    <?php
    //在 config.txt 中设置各种信息
    $fd = fopen ("config.txt", "r");
    $buffer = fgets($fd, 4096);
    $width= intval($buffer);
    $buffer = fgets($fd, 4096);
    $height= intval($buffer);
    $buffer = fgets($fd, 4096);
    $maxvalue = intval($buffer);
    $buffer = fgets($fd, 4096);
    $curvalue = intval($buffer);
    $percentfull=$curvalue/$maxvalue * 100;
    $buffer = fgets($fd, 4096);
    $notchcount= intval($buffer);
    //读入刻度设置
    for ($i = 0; $i <= $notchcount; $i++) {
        $buffer = fgets($fd, 4096);
        $notches[$i] =substr($buffer,0,strlen($buffer)-1);
    }
    fclose ($fd);
    Header("Content-type: image/png");
    $im = imagecreate($width,$height);
    $white=ImageColorAllocate($im,255,255,255);
```

```
    $black=ImageColorAllocate($im,0,0,0);
    //读入初始化图像
    $im_bottom = imagecreatefrompng ("bottom.png");
    $im_middle_empty = imagecreatefrompng("middle_empty.png");
    $im_middle_full = imagecreatefrompng("middle_full.png");
    $im_top_full = imagecreatefrompng("top_full.png");
    $im_top_empty = imagecreatefrompng("top_empty.png");
    $im_notch_empty = imagecreatefrompng("notch_empty.png");
    $im_notch_full = imagecreatefrompng("notch_full.png");
    $im_liquid_top =  imagecreatefrompng("liquid_top.png");
    $white = ImageColorAllocate ($im_notch_full, 255, 255, 255);
    ImageColorTransparent($im_notch_full,$white);

    $white = ImageColorAllocate ($im_notch_empty, 255, 255, 255);
    ImageColorTransparent($im_notch_empty,$white);
    $white = ImageColorAllocate ($im_liquid_top, 255, 255, 255);
    ImageColorTransparent($im_liquid_top,$white);

    //Puts the bottom bulb on the thermometer
    ImageCopyResized($im,$im_bottom,0,$height-imagesy($im_bottom),0,0,imagesx
($im_bottom),imagesy($im_bottom),imagesx($im_bottom),imagesy($im_bottom));
    if ($percentfull>100){$percentfull=100;}
    if ($percentfull<0){$percentfull=0;}

    //更改图像大小
    // if ($percentfull < 100){
    ImageCopyResized($im,$im_top_empty,0,0,0,0,imagesx($im_top_empty),imagesy
($im_top_empty),imagesx($im_top_empty),imagesy($im_top_empty));
    // }

    //显示温度计没有显示的部分
    $tubeheight=$height-imagesy($im_top_empty) - imagesy($im_bottom) + 1 ;
    $emptyheight=floor($tubeheight * ((100-$percentfull)/100)) ;
    ImageCopyResized($im,$im_middle_empty,0,imagesy($im_top_empty),
                     0,0,imagesx($im_middle_empty),$emptyheight ,
                     imagesx($im_middle_empty),imagesy($im_middle_empty));
    //显示所有刻度
    for ($i = 0; $i <= $notchcount; $i++) {
        ImageCopyResized($im,$im_notch_empty,0,imagesy($im_top_empty) +
($tubeheight)*($i/$notchcount)-5,
            0,0,imagesx($im_notch_empty),imagesy($im_notch_empty),
            imagesx($im_notch_empty),imagesy($im_notch_empty));
    }

    //Puts in the full part of the tube it
    ImageCopyResized($im,$im_middle_full,0,imagesy($im_top_empty) + $emptyheight,
                     0,0,imagesx($im_middle_full),$tubeheight-$emptyheight ,
                     imagesx($im_middle_full),imagesy($im_middle_full));
    //显示刻度顶部
    ImageCopyResized($im,$im_liquid_top,0,imagesy($im_top_empty)+$emptyheight
-5,
                     0,0,imagesx( $im_liquid_top),imagesy( $im_liquid_top) ,
```

```
                    imagesx( $im_liquid_top),imagesy( $im_liquid_top));
    for ($i = 0; $i <= $notchcount; $i++) {
    ImageCopyResized($im,$im_notch_full,0,imagesy($im_top_empty)+($tubeheight
)* ($i/$notchcount)-5,
            0,0,imagesx($im_notch_empty),imagesy($im_notch_empty),
            imagesx($im_notch_empty),imagesy($im_notch_empty));
    ImageString($im,5,50, imagesy($im_top_empty) + ($tubeheight)*
($i/$notchcount)-8,$notches[$i] ,$black);

    }

    imagePng($im);
    ImageDestroy($im);
    imagedestroy($im_bottom);
    ?>
```

执行结果如图 10-13 所示。

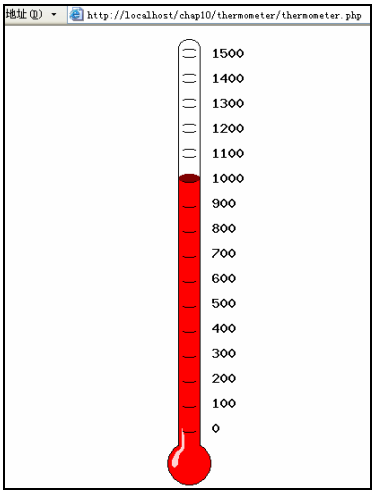


图 10-13　温度计的显示结果

实例 10-12　将文字添加到图片中

本实例演示如何将文字添加到现存的图片中，由两个文件组成。index.php 文件用于设置文本，如代码 10-14 所示。首先读入图片并将其作为背景，然后调用显示图像的 img.php 文件并显示文件，如代码 10-15 所示。

【代码 10-14】　index.php

```
    <html>

    <head>
    <meta http-equiv="Content-Language" content="en-us">
    <meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
    <meta name="GENERATOR" content="Microsoft FrontPage 4.0">
    <meta name="ProgId" content="FrontPage.Editor.Document">
    <title>Free Script - DEMO Add caption to your image</title>
    </head>
```

```
    <body style="font-family: Arial; font-size: 10pt">
    将文字添加到图形上面
    <form method="POST" action="index.php">
    <input type="text" name="text" size="20" maxlength="25"><br>
      <input type="submit" value="提交文字" name="submit">
    </form>
    <br>
    <?php
    $imgfile="smp.jpg";

    if (!empty($_POST['text']))
        $text=$_POST['text'];
    else
        $text="";
    if($text!="")
    {
        print "<img border='0' src='img.php?imgfile=smp.jpg&text=$text' width=
'200' height='200'>";
    }
    else
    {
        echo "<img border='0' src='smp.jpg' width='200' height='200'>";

    }
    ?>
    </body>
    </html>
```

【代码 10-15】 img.php

```
    <?php
    $font = 'arial.ttf';
    $imgfile=$_GET['imgfile'];
    $ext=substr($imgfile,-3);
    $ext=strtolower($ext);
    if($ext=="jpg" || $ext=="jpe") $im=@imagecreatefromjpeg("$imgfile");
    elseif ($ext=="gif") $im=@imagecreatefromgif("$imgfile");
    else {print "Unknown image format"; exit;}

     if (!$im) { /* See if it failed */
          $im = ImageCreate (150, 30); /* Create a blank image */
          $bgc = ImageColorAllocate ($im, 255, 255, 255);
          $tc  = ImageColorAllocate ($im, 0, 0, 0);
          ImageFilledRectangle ($im, 0, 0, 150, 30, $bgc);
          /* Output an errmsg */
          ImageString($im, 1, 5, 5, "Error loading $imgfile", $tc);
        return $im;
       }

    $x=imagesx($im);
    $y=imagesy($im);
    $fontsize=$x/20;
```

```
$fontsize=floor($fontsize);
if($fontsize<10) $fontsize=10;

$white = imagecolorallocate($im, 255, 255, 255);
$black = imagecolorallocate($im, 0, 0, 0);

imagettftext($im,$fontsize, 0, 12, $fontsize+8, $black, $font, $_GET['text']);
imagettftext($im,$fontsize, 0, 10, $fontsize+6, $white, $font, $_GET['text']);
if($ext=="gif")
    {
    header("Content-type: image/gif");
    imageGIF($im);
    }
else
    {
    header("Content-type: image/jpeg");
    imagejpeg($im);
    }
imagedetroy($im);
?>
```

执行结果如图 10-14 所示。



图 10-14　将文字添加到图像

实例 10-13　改变图像大小

本实例可以将一幅图像按照指定的比例进行缩放，如代码 10-16 所示。

【代码 10-16】　resizer.php

```
<?php
// 文件名以及显示图像的比例
$imgfile = 'smp.jpg';
$percent = 0.5;  //为原来图像的一半大小
header('Content-type: image/jpeg');

list($width, $height) = getimagesize($imgfile);
$newwidth = $width * $percent;
```

```
    $newheight = $height * $percent;

    $thumb = ImageCreateTrueColor($newwidth,$newheight);
    $source = imagecreatefromjpeg($imgfile);

    imagecopyresized($thumb, $source, 0, 0, 0, 0, $newwidth, $newheight, $width,
$height);

    imagejpeg($thumb);
    ?>
```

执行结果如图 10-15 所示。



图 10-15　图像的缩放

实例 10-14　绘制销售报表

本实例用于绘制销售报表。根据用户输入数据的不同，将会显示不同的数据报表，如代码 10-17 所示。

【代码 10-17】　sales.php

```
    <?php
       if (!empty($_POST['Submit'])){
          // read the post data
          $data = array($_POST['T1'],$_POST['T2'],$_POST['T3'], $_POST['T4'],
$_POST['T5']);
          $x_fld = array('Jan','Feb','March','April','May');
          $max = 0;
          for ($i=0;$i<5;$i++){
            if ($data[$i] > $max)$max=$data[$i];  // find the largest data
          }
          $im = imagecreate(320,255); // width , height px
          $white = imagecolorallocate($im,255,255,255); // allocate some color
from RGB components remeber Physics
          $black = imagecolorallocate($im,0,0,0);   //
          $red = imagecolorallocate($im,255,0,0);   //
          $green = imagecolorallocate($im,0,255,0); //
          $blue = imagecolorallocate($im,0,0,255);  //
          //
          // create background box
          //imagerectangle($im, 1, 1, 319, 239, $black);
          //draw X, Y Co-Ordinate
          imageline($im, 10, 5, 10, 230, $blue );
          imageline($im, 10, 230, 300, 230, $blue );
          //Print X, Y
```

```
          imagestring($im,3,8,2,"Y",$black);
          imagestring($im,3,305,222,"X",$black);

          // what next draw the bars
          $x = 15;   // bar x1 position
          $y = 230;   // bar $y1 position
          $x_width = 20;  // width of bars
          $y_ht = 0; // height of bars, will be calculated later
          // get into some meat now, cheese for vegetarians;
          for ($i=0;$i<5;$i++){
            $y_ht = ($data[$i]/$max)* 100;
            imagerectangle($im,$x,$y,$x+$x_width,($y-$y_ht),$red);
            imagestring( $im,2,$x-1,$y+1,$x_fld[$i],$black);
            imagestring( $im,2,$x-1,$y+10,$data[$i],$black);
            $x += ($x_width+20);   // 20 is diff between two bars;

          }
          imagepng($im, "mypic.png");

          echo "<img src='mypic.png'><p></p>";
           imagedestroy($im);
       }
    ?>
    <html>
    <head>
    <meta http-equiv="Content-Language" content="en-us">
    <title>Sales Report</title>
    </head>
    <body>
    <form method="POST" name="myForm" action="sales.php">
      <div align="left">
        <table border="1" cellpadding="0" cellspacing="0" style="border-collapse:
collapse" bordercolor="#111111" width="454" height="159">
          <tr>
            <td width="454" height="32" colspan="2" bgcolor="#CCCCCC">
            <p align="left"><b><font face="Tahoma">销售报表:</font></b></td>
          </tr>
          <tr>
            <td width="207" height="41">
            <p align="right"><b><font face="Tahoma" size="2">1 月:</font></b></td>
            <td width="241" height="41"><input type="text" name="T1" size="6"></td>
          </tr>
          <tr>
            <td width="207" height="44">
            <p align="right"><b><font face="Tahoma" size="2">2 月:</font></b></td>
            <td width="241" height="44"><input type="text" name="T2" size="6"></td>
          </tr>
          <tr>
            <td width="207" height="35" align="right"><b>
            <font face="Tahoma" size="2">3 月:</font></b></td>
            <td width="241" height="35"><input type="text" name="T3" size="6"></td>
          </tr>
```

```
        <tr>
          <td width="207" height="35" align="right"><b>
          <font face="Tahoma" size="2">4 月:</font></b></td>
          <td width="241" height="35"><input type="text" name="T4" size="6"></td>
        </tr>
        <tr>
          <td width="207" height="35" align="right"><b>
          <font face="Tahoma" size="2">5 月:</font></b></td>
         <td width="241" height="35"><input type="text" name="T5" size="6"></td>
        </tr>
        <tr>
          <td width="448" height="35" align="right" colspan="2">
          <p align="center"><font face="Tahoma">
          <input type="submit" value="提交" name="Submit"></font></td>
        </tr>
      </table>
    </div>
  </form>
  <?php
  ?>
  </body>
  </html>
```

执行结果如图 10-16 所示。

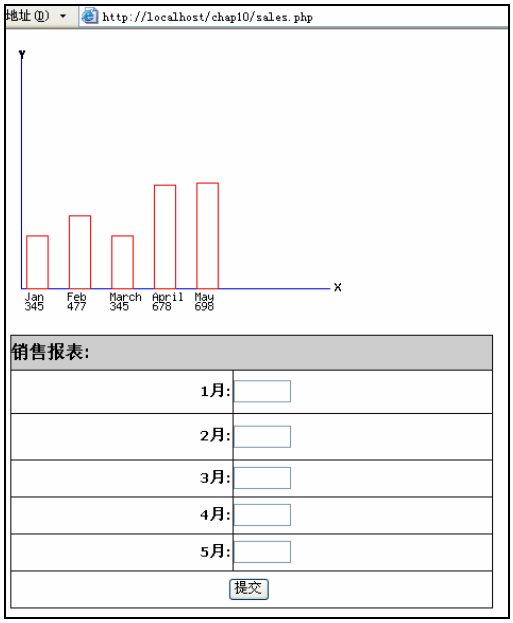


图 10-16　销售报表

细心的读者会发现，如果在创建的图像上显示中文，则会显示乱码。其实 PHP 中的 GD 库是支持中文的，但是必须要以 UTF-8 格式的参数来传递。可以实验下面的小程序。

实例 10-15　显示中文

本实例将演示如何在图像上面显示中文，如代码 10-18 所示。

【代码 10-18】　showchinese.php

```
<?php
Header("Content-type: image/png");
$im = imagecreate(100,100);
$bkg = ImageColorAllocate($im, 0,0,0);
$clr = ImageColorAllocate($im, 255,255,255);
// 不同系统操作系统路径不一样
$fnt = "c:/windows/fonts/simhei.ttf";

$str = chr(0xE4).chr(0xB8).chr(0xAD).chr(0xE6).chr(0x96).chr(0x87)."OK";
// "中文 OK" in UTF-8
ImageTTFText($im, 20, 0, 10, 30, $clr, $fnt, $str);
Imagepng($im);
ImageDestroy($im);
?>
```

执行结果如图 10-17 所示。



图 10-17　中文显示结果

如果仅仅是要显示一个固定的中文字符串，那么可以在 Word、金山等软件中将要显示的字符转换为 UTF-8 格式，再粘贴到程序中就可以了。

如果要动态地根据用户的输入来显示相应的中文字符，那么就必须要解决将汉字的内码转化为 UTF-8 格式的问题。

关于 GBK-Unicode 的转换，笔者使用了一种笨办法，在这里写出来供读者借鉴。

先用下面一段小程序把所有的 GBK 字符显示出来，然后在 Word 中将它转换为 Unicode。依据这个字符串把每一个字符转成为相应的 Unicode 值。Unicode 表示为 UTF-8 的过程参照。Unicode 上的转换程序为 cvtutf.c 和 cvtutf.h。

```
<?php
for($i = 0x81;  $i <= 0xFE; $i++){
    // 0x81 - 0xFE
    for($j = 0x40; $j <= 0xFE; $j++){
    // 0x40 - 0x7E 0x80 - 0xFE
        if($j != 0x7F){
            echo chr($i).chr($j);
        }
    }
}
?>
```

相比其他的网络编程语言，PHP 在图形绘制方面有着极大的优势，本书已经列举了大量的案例，读者可以自己参照案例学习和自行发挥。

第11章　会 话 控 制

会话控制是 PHP 中的一项重要内容，通过会话控制，很多任务都能够轻松的解决。

会话控制的思想是指能够在网站中根据一个会话跟踪用户。如果做到了这点，就可以比较容易地做到对用户登录的支持，并且根据其授权级别和个人喜好显示相应的内容。电子商务网站中的购物车就是一个很好的例子。

PHP 的会话是通过惟一的会话 ID 来驱动的，会话 ID 是一个加密的随机数字。它由 PHP 生成，在会话的生命周期中都会保存在客户端。它可以保存在机器的 Cookie 中，也可以通过 URL 在网络上传递。

会话 ID 就像是一个钥匙，允许注册一些特定的变量，也被称为会话变量。这些变量的内容保存在服务器端。会话 ID 是客户端惟一可见的信息。如果在一次特定的网络连接中，客户端通过 Cookie 或者 URL 看到会话 ID，那么就可以访问该会话保存在服务器上的会话变量。在默认的情况下，会话变量保存在服务器中的普通文件中。

11.1　Cookie 的应用

11.1.1　Cookie 的概念

Cookie 的中文意思是甜饼。Cookie 其实就是一小段信息，它可以由脚本在客户端机器上保存。从本质上讲，它是客户的身份证。它不能作为代码执行，也不会传送病毒。它为连接专有，并且只能由提供它的服务器来读取。

它记录了客户的用户 ID、密码、浏览过的网页和停留的时间等信息，当再次来到该网站时，网站通过读取 Cookie，便可得知相关信息，做出相应的动作。比如在页面显示欢迎你的标语，或者让用户不输入 ID、密码就直接登录等。可以在 IE 的“工具/Internet 选项”的“常规”选项卡中，选择“设置/查看文件”，查看所有保存到电脑里的 Cookie，如图 11-1 所示。这些文件通常是以 user@domain 格式命名的，user 是本地用户名，domain 是所访问的网站的域名。

为了保证上网安全，需要对 Cookie 进行适当设置。打开“工具/Internet 选项”中的“隐私”选项卡（注意该设置只在 IE6.0 中存在，其他版本 IE 可以在“工具/Internet 选项”的“安全”标签中单击“自定义级别”按钮，进行简单调整），调整 Cookie 的安全级别。

通常情况，可以将滑块调整到“中高”或者“高”的位置。多数的论坛站点需要使用 Cookie 信息，如果从来不去这些地方，可以将安全级调到“阻止所有 Cookies”。如果只是为了禁止个别网站的 Cookie，可以单击“站点”按钮，将要屏蔽的网站添加到列表中。

Cookie 中的内容大多数经过了加密处理，因此看起来只是一些毫无意义的字母和数字的组合，只有服务器的 CGI 处理程序才知道它们真正的含义。

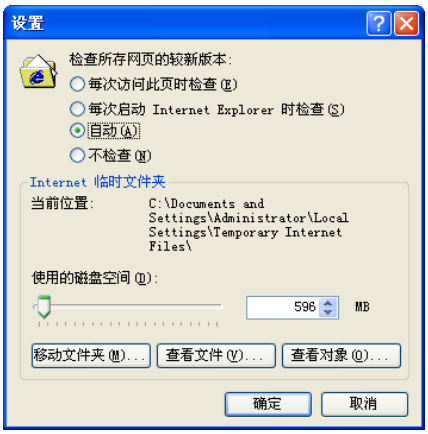


图 11-1　Cookie 设置

当在浏览器地址栏中键入了一个 Web 站点的 URL，浏览器会向该 Web 站点发送一个读取网页的请求，并将结果在显示器上显示。这时该网页在电脑上寻找网站设置的 Cookie 文件，如果找到，浏览器会把 Cookie 文件中的数据连同前面输入的 URL 一同发送到服务器。服务器收到 Cookie 数据，就会在数据库中检索出用户的 ID，购物记录、个人喜好等信息，并记录下新的内容，增加到数据库和 Cookie 文件中去。如果没有检测到 Cookie 或者 Cookie 信息与数据库中的信息不符合，则说明是第一次浏览该网站，服务器的 CGI 程序将创建新的 ID 信息，并保存到数据库中。

Cookie 是利用了网页代码中的 HTTP 头信息进行传递的，浏览器的每一次网页请求，都可以伴随 Cookie 传递，例如，浏览器的打开或刷新网页操作。Cookie 在电脑上保存的时间是不一样的，这些都是由服务器的设置不同决定的。Cookie 有一个 Expires（有效期）属性，这个属性决定了 Cookie 的保存时间，服务器可以通过设定 Expires 字段的数值，来改变 Cookie 的保存时间。如果不设置该属性，那么 Cookie 只在浏览网页期间有效，关闭浏览器，Cookie 会自动消失，绝大多数网站属于这种情况。通常情况下，Cookie 包含 Server、Expires、Name、Value 几个字段，其中对服务器有用的只是 Name 和 Value 字段，Expires 等字段的内容仅仅是为了告诉浏览器如何处理这些 Cookies。

11.1.2　PHP 中的 Cookie 编程

PHP 透明地支持 HTTP Cookie。可以用 setcookie()或 setrawcookie()函数来设置 Cookie。Cookie 是 HTTP 标头的一部分，因此 setcookie()函数必须在其他信息被输出到浏览器前调用，这和对 header()函数的限制类似。可以使用输出缓冲函数来延迟脚本的输出，直到按需要设置好了所有的 Cookie 或者其他 HTTP 头。

如果 variables_order 中包括“C”，则任何从客户端发送的 Cookie 都会被自动包括进 $_COOKIE 全局数组。如果希望对一个 Cookie 变量设置多个值，则需在 Cookie 的名称后加[]符号。

1．setcookie 函数

setcookie 函数用于手动地设置 Cookie，该函数声明如下：

```
    bool setcookie ( string name [, string value [, int expire [, string path [,
string domain [, bool secure]]]]] )
```

setcookie()定义一个和其余的 HTTP 头一起发送的 cookie。和其他 HTTP 头一样，cookie 必须在脚本的任何其他输出之前发送（这是协议限制）。这需要将函数的调用放到任何输出之前，包括<html>和<head>标签以及任何空格。如果在调用 setcookie()之前有任何输出，本函数将失败并返回 FALSE。如果 setcookie()函数成功运行，将返回 TRUE。这并不说明用户接受了 cookie。

setcookie 的参数除了 name 外，其他都是可选的。可以用空字符串（""）替换某参数以跳过该参数。因为参数 expire 是整型，不能用空字符串跳过，可以用零（0）来代替。表 11-1 所示对 setcookie()的每一个参数都进行了解释。

表 11-1 setcookie()参数详解

参 数	说 明	应 用
name	Cookie 的名字	使用$_COOKIE['cookiename']调用名为 cookiename 的 cookie。
value	Cookie 的值。此值保存在客户端，不能用来保存敏感数据	假定 name 是'cookiename'，可以通过 $_COOKIE['cookiename'] 取得其值
expire	Cookie 过期的时间	time()+60*60*24*30 将设定 cookie 30 天后失效。如果未设定，cookie 将会在会话结束后（一般是浏览器关闭）失效
path	Cookie 在服务器端的有效路径	如果该参数设为'/' ，cookie 就在整个 domain 内有效，如果设为'/foo/'，cookie，就只在 domain 下的/foo/目录及其子目录内有效，例如/foo/bar/。默认值为设定 cookie 的当前目录
domain	该 cookie 有效的域名	要使 cookie 能在如 example.com 域名下的所有子域都有效，该参数应该设为'.example.com'。如果该参数设为 www.example.com，就只在 www 子域内有效
secure	指明 cookie 是否仅通过安全的 HTTPS 连接传送。当设成 TRUE 时，cookie 仅在安全的连接中被设置。默认值为 FALSE	0 或 1

当 cookie 被设置后，便可以在其他页面通过$_COOKIE 或$HTTP_COOKIE_VARS 数组取得其值。

在使用 cookie 时有以下注意事项。

cookies 不会在设置它的本页生效，要测试一个 cookie 是否被成功的设定，可以在其到期之前通过另外一个页面来访问其值。过期时间是通过参数 expire 来设置的。可以简单地使用 print_r($_COOKIE)；来调试现有的 cookies。

Cookie 必须用和设定时相同的参数才能删除。如果其值为空字符串，或者是 FALSE，并且其他的参数都和前一次调用 setcookie 时相同，那么所指定名称的 cookie 将会在远程客户端被删除。

由于把 cookie 的值设为 FALSE 会使客户端尝试删除这个 cookie，所以要在 cookie 上保存 TRUE 或 FALSE 时不应该直接使用 boolean 值，而应该用 0 来表示 FALSE，用 1 来表示 TRUE 可以把 cookie 的名称设置成一个数组，但是数组 cookie 中的每个元素的值将会被单独保存在用户的系统中。考虑使用 explode()函数用多个名称和值设定一个 cookie。

实例 11-1 setcookie 函数的应用

本实例演示了如何创建 cookie 并在第 2 个页面输出 cookie 值。实例由 2 个文件组成，第 1 个文件名为 setcookie.php，用来设置 cookie，如代码 11-1 所示。第 2 个文件名为 testcookie.php，用来显示在第 1 个页面中设置的 cookie 值。

【代码 11-1】　setcookie.php

```
<?php
$value = 'i like php';
$value1= 'i hate php';

setcookie("TestCookie", $value);  //设置 cookie 名为 testcookie
setcookie("TestCookie", $value,time()+3600);  /* 1 小时候失效 */
setcookie("TestCookie1", $value1);  //设置 cookie 名为 testcookie1
//为了显示这些 cookie 的值，必须在其他页面输出
?>
<a href="testcookie.php">输出 cookie</a>
```

执行结果如图 11-2 所示。



图 11-2　setcookie.php 的执行结果

在图 11-2 中，单击链接，可以进入到显示 cookie 的页面，即 testcookie.php。

【代码 11-2】　testcookie.php

```
<?php
// 输出单独的 cookie
echo $_COOKIE["TestCookie"] . "<br>";

// 另一个调试的方法就是输出所有的 cookie
print_r($_COOKIE);
?>
```

执行结果如图 11-3 所示。

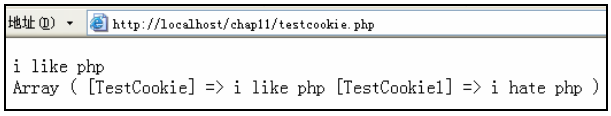


图 11-3　显示 cookie 的结果

从上面的例子可以看出，实际上 cookie 已经跟踪了用户的状态，即能够在第 2 个页面得到用户在第 1 个页面留下的状态信息。

既然能够创建 cookie，系统也提供了删除 cookie 的方法。要删除一个 cookie，还是使用 setcookie 函数，只是设置其过期的时间为过去的时间，从而强迫 cookie 过期失效。比如要让 setcookie.php 中的 cookie 失效，可以采用 deletecookie.php 中的代码。

实例 11-2　删除 cookie

本实例演示了如何删除 cookie，如代码 11-3 所示。

【代码 11-3】　deletecookie.php

```
<?php
```

```
setcookie("TestCookie", "", time() - 3600);

//输出 testcookie
if (!empty($_COOKIE["TestCookie"]))
    echo "testcookie 值为：".$_COOKIE["TestCookie"] . "<br>";
else
    echo "testcookie1 被注销。<br>";
//输出 testcookie1
echo "testcookie1 值为：".$_COOKIE["TestCookie1"] . "<br>";
//输出所有 cookie
print_r($_COOKIE);
?>
```

执行结果如图 11-4 所示。



图 11-4　删除 cookie

要在运行实例 11-1 后，才能运行该实例来查看效果。从图 11-4 中可以看出，名称为 testcookie 的 cookie 已经被注销，而名称为 testcookie1 的 cookie 还存在。

提示　当一个 Cookie 被删除时，它的值在当前页仍然有效，因此必须刷新后才能生效。

实例 11-3　创建 cookie 数组

本实例演示如何创建一个 cookie 数组，如代码 11-4 所示。

【代码 11-4】　cookiearray.php

```
<?php
// 设定 cookie
setcookie("cookie[three]", "i am three");
setcookie("cookie[two]", "i am two");
setcookie("cookie[one]", "i am one");

// 刷新页面后，显示出来
if (isset($_COOKIE['cookie'])) {
    foreach ($_COOKIE['cookie'] as $name => $value) {
        echo "$name : $value <br />\n";
    }
}
?>
```

执行结果如图 11-5 所示。

2．Header 函数

在 PHP 中提供了一个 Header()函数，在图像处理一章，已经简单使用了该函数。该函数提供了设置 HTTP 报头信息的一种方法。通过 Header()函数，可以方便地处理 Cookie。该函数的声明如下：

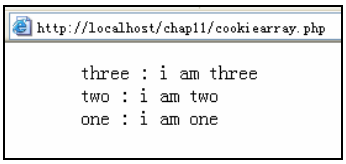


图 11-5　cookie 数组

```
void header ( string string [, bool replace [, int http_response_code]] )
```

header()函数用来发送一个原始 HTTP 标头。

可选参数 replace 指明是替换掉前一条类似的标头还是增加一条相同类型的标头。默认为替换，但如果将其设为 FALSE 则可以强制发送多个同类标头。例如：

```
<?php
header('WWW-Authenticate: Negotiate');
header('WWW-Authenticate: NTLM', false);
?>
```

第 2 个可选参数 http_response_code 强制将 HTTP 响应代码设为指定值。

有两种特殊的 header 调用。第 1 种是标头以字符串“HTTP/”（大小写不重要）开头的，可以用来确定要发送的 HTTP 状态码。例如，如果配置了 Apache 利用 PHP 来处理找不到文件的错误请求（使用 ErrorDocument 指令），需要确保脚本产生了正确的状态码。如下所示：

```
<?php
header("HTTP/1.0 404 Not Found")
?>
```



注意　HTTP 状态码标头行总是第 1 个被发送到客户端，而并不管实际的 header()调用是否是第 1 个。除非 HTTP 标头已经发送出去，任何时候都可以通过用新的状态行调用 header()函数来覆盖原先的。

第 2 种特殊情况是以“Location:”标头。它不只是把这个标头发送回浏览器，它还将一个 REDIRECT（302）状态码返回给浏览器，除非之前已经发出了某个 3xx 状态码，如下所示：

```
<?php
header("Location: http://www.example.com/"); /* 重定向浏览器 */
/* 确保重定向后，后续代码不会被执行 */
exit;
?>
```

HTTP/1.1 标准需要一个绝对地址的 URI 做为“Location:”的参数，但有一些客户端支持相对 URI。通常可以使用$_SERVER['HTTP_HOST']、$_SERVER['PHP_SELF']及 dirname()函数来从相对的 URI 产生出绝对 URI，如下所示：

```
<?php
header("Location: http://".$_SERVER['HTTP_HOST']
                   . rtrim(dirname($_SERVER['PHP_SELF']), '/\\')
                  ."/".$relative_url);
?>
```



提示　即使启用了 session.use_trans_sid，Session ID 也不会随着 Location 头信息被传递。必须手工用 SID 常量来传递。

PHP 脚本通常会产生一些动态内容，这些内容必须不被浏览器或代理服务器缓存。很多

代理服务器和浏览器都可以被下面的方法禁止缓存，如下所示：

```
<?php
header("Cache-Control: no-cache, must-revalidate"); // HTTP/1.1
header("Expires: Mon, 26 Jul 1997 05:00:00 GMT"); // 过去的时间
?>
```



读者可能会发现，即使使用上面所有的代码，网页也没有被缓冲。用户有很多选项可以设置来改变浏览器的默认缓存行为。通过发送上述标头，可以覆盖任何导致脚本页面被缓存的设置。

另外，当使用了 session 时，利用 session_cache_limiter()函数和 session.cache_limiter 选项可以自动产生正确的缓存相关标头。

要记住 header()必须在任何实际输出之前调用，不论是来自普通的 HTML 标记，空行或者 PHP。有一个常见错误就是在通过 include()，require()或一些其他的文件存取类函数，读取代码时，有一些空格或者空行在调用 header()之前被发送了出去。同样，在一个单独的 PHP/HTML 文件中这个错误也很普遍，如下所示：

```
<html>
<?php
/* 这将产生一个错误，因为在调用 header()
* 之前已经输出了东西 */
header('Location: http://www.example.com/');
?>
```



PHP 4 及其后续版本，可以通过一些输出缓冲函数来解决这个问题。代价是把所有向浏览器的输出都缓存在服务器，直到下命令发送它们。可以在代码中使用 ob_start()及 ob_end_flush()来实现这样的功能，或者通过修改 php.ini 中的 output_buffering 配置选项来实现，也可以通过修改服务器配置文件来实现。

前面简单介绍了 Header()函数的应用和注意事项，接下来可以通过 Header()函数来发送 Cookie 了。

实例 11-4　通过 cookie 跟踪用户访问次数

本页面通过 header 函数发送 cookie，来跟踪用户访问某个服务器的次数，如代码 11-5 所示。

【代码 11-5】　Headcookie.php

```
<?php
if  (!empty($_cookie['tracker']))
    $tracker=++$_cookie['tracker'];
else
    $tracker=1;

//设置失效时间
$expires_soon=date("1,d-m-y h:i:s",time()+54000);

//发送 header
header("set-cookie: tracker=$tracker;expires=$expires_soon;");
?>
```

```
<html>
<head>
<title>header 传送 cookie</title>
</head>
</body>
<?php
echo "你已经访问本页 $tracker 次了！";
?>
</body>
</html>
```

通过以上程序，服务器就可以跟踪客户访问该页面的次数了。

11.2　实现会话控制

前面提到的 Cookie 虽然可以在客户端保存一定数量的会话状态，但是事实上全部采用 Cookie 来解决会话控制是不现实的，因为 Cookie 本身的容量很有限。因此这里提供另外一种解决方案，那就是只在客户端保存一个会话标志符，然后将会话数据都存储在服务器上或者数据库中。PHP 中提供了一套用于会话的函数，而且这些函数给出了决定会话数据存储在何处所需要的控制等级。在某些情况下，也可以使用数据库。

11.2.1　会话的基本方式

通过 PHP 进行会话处理的时候，有几个问题是经常碰到的。比如多个 PHP 应用程序服务器使用同一个后台数据库的时候，会话如何工作；如何安全地生成一个会话标志符，即会话 ID；如果客户端限制了 Cookie，如何传送会话 ID 等。

1．会话 ID 的传送

传输会话 ID 常用的有两种方式，一种就是 Cookie 方式，而另外一种是 URL 方式。

（1）cookie 方式传送。使用 cookie 传送会话 ID 是最为简单的。但是有些客户可能限制使用 cookie。如果在客户限制 cookie 的条件下，仍然需要正常工作，则必须要通过其他方法实现。

（2）URL 传送。在这种方式中，URL 本身用来传送会话。会话标志被简单地附加在 URL 的尾部，或者作为窗体中的一个变量来传递。虽然该方法使用起来不方便，但是很多时候还是非常重要的。

2．会话 ID 的生成

PHP 的会话函数会自动处理会话 ID 的创建。但是仍然可以通过手工方法来创建会话 ID。会话 ID 必须是不容易被人猜出来的，否则就会有安全隐患。

一般推荐生成会话 ID 使用随机数发生器函数 srand，如下所示：

```
srand((double)microtime()*1000000);
```

在调用以上函数之后，要想生成一个惟一的会话 ID，还必须调用以下语句：

```
md5(uniqid(rand()));
```

当然最安全的方法是让 PHP 自己来生成会话 ID。

11.2.2　创建会话

实现一个简单的会话一般要经过如下步骤：启动一个会话，注册会话变量，使用会话变

量，注销会话变量。

1．启动一个会话

在 PHP 中提供了两种方法来创建一个会话。

（1）session_start()方法。session_start 是一个函数，如果要创建一个会话，只需要在写其他代码之前首先调用该函数，如下所示：

```
session_start();
```

该函数的声明如下：

```
bool session_start ( void )
```

函数首先判断是否有一个会话 ID 存在，如果不存在，就创建一个，并且使其能够通过全局数组$_SESSION 进行访问。如果已经存在，则将这个已经注册的会话变量载入以供使用。

（2）通过设置 php.ini，自动创建会话。如果不想在程序代码中使用“session_start();”语句，则还有一种比较方便的方法，就是当用户访问网站的时候自动启动一个会话。只需要将 php.ini 文件中的 session.auto_start 选项激活即可。但是这种方法有一个缺陷，启动 auto_start 设置会导致无法使用对象作为会话变量。

2．创建会话变量

启动会话后，会话变量全部保存在数组$_SESSION 中，同时也保存在$HTTP_SESSION_VARS 中，但是建议保存在$_SESSION 中。通过$_SESSION 创建一个会话变量非常容易，只需要直接给该数组添加一个元素即可，如下所示：

```
$_SESSION['sesssion_name']=session_value;
```

经过以上设置后，会话变量会在有效期内跟踪用户。

3．使用会话变量

设置好会话变量之后，接下来就可以使用了，如要获取一个会话变量的值，使用如下语句：

```
if (!empty($_SESSION['sesssion_name']))
    $myValue=$_SESSION['sesssion_name'];
```

4．注销会话变量

既然有创建会话变量，肯定能够注销该会话变量。该操作如同操作数组一样，直接注销$_SESSION 数组的某个元素即可，如下所示：

如要注销$_SESSION['sesssion_name']变量，可以使用如下语句：

```
unset($_SESSION['sesssion_name']);
```

不能一次注销整个$_SESSION 数组，这样会禁止整个会话功能。如果要一次性注销所有的会话变量，则可以使用如下语句，将一个空的数组赋值给$_SESSION：

```
$_SESSION=array();
```

如果整个会话已经结束，首先应该注销所有的会话变量，然后使用 session_destroy()函数来清除会话 ID，如下所示：

```
session_destroy();
```

11.2.3 会话应用实例

下面通过一些实例来学习会话的应用。

1．会话控制简单应用

实例 11-5 session 应用实例

本实例比较简单，主要是在第 1 个页面设置几个变量，在第 2 个页面输出这些变量，在

第 3 个页面中注销会话变量，然后清除会话 ID。第 1 个页面文件名为 session_1.php，如代码 11-6 所示。

【代码 11-6】　session_1.php

```
<?php
//session_1.php
session_start();  //启动
echo '欢迎来到本页';
$_SESSION['favcolor'] = 'green';
$_SESSION['animal']   = 'cat';
$_SESSION['time']     = time();

// 设置链接，进入到第 2 页
echo '<br /><a href="session_2.php">第 2 页 </a>';

?>
```

执行结果如图 11-6 所示。

在图中单击“第 2 页”链接，可以进入到 session_2.php 文件，如代码 11-7 所示。

【代码 11-7】　session_2.php

```
<?php
//session_2.php
session_start();  //启动
echo '欢迎到第 2 页<br />';

echo  $_SESSION['favcolor']."<br>"; // 输出 session
echo $_SESSION['animal']."<br>";   // 输出 session
echo date('Y m d H:i:s', $_SESSION['time']);

echo '<br /><a href="session_3.php">第 3 页</a>';
?>
```

输出结果如图 11-7 所示。



图 11-6　session_1.php 执行结果



图 11-7　显示会话变量的值

在图 11-7 中，输出了在 session_1.php 文件设置的一些会话变量值。接着单击“第 3 页”进入到 session_3.php 文件，如代码 11-8 所示。

【代码 11-8】　session_3.php

```
<?php
//session_2.php
session_start();  //启动
echo '欢迎到第 3 页<br />';
```

```
unset($_SESSION['favcolor']);
if (!empty($_SESSION['favcolor']))
echo  "SESSION['favcolor']的值是：".$_SESSION['favcolor']."<br>";//输出 session
else
    echo "SESSION['favcolor']的值被删除了!";
session_destroy();  //注销会话 ID

echo '<br /><a href="session_1.php">第 1 页</a>';
?>
```

session_3.php 的执行结果如图 11-8 所示。



图 11-8　注销 session 变量

从图 11-8 的结果可以看出，如果调用 unset 函数注销了一个会话变量，则输出时为空。

2．身份验证

身份验证是会话的一个重要功能。一旦用户登录成功后，就会通过会话 ID 一直跟踪用户，并不需要用户在每个页面输入身份验证信息。但是如果用户登录失败，却要尝试登录网站的其他页面，则会提示没有登录，要求重新登录。关于身份验证的应用，本书将在后面的“登录”模块中进行介绍。

3．购物车

购物车也是会话的一个重要功能。通常人们在进行购物的时候，总是选购多样物品，然后一次性支付。在挑选的时候，总是需要将商品临时放在一个购物篮中，用户可以方便地添加或者删除商品。而这个购物车的实现，就是通过会话来进行的。

11.2.4　会话控制的配置

在 php.ini 文件中，提供了一组会话配置选项，下面列出一些常用的设置。

Session.save_handler = files：用于保存 Session 变量，默认情况下用文件。

Session.save_path = c:/temp：保存 Session 变量的目录，在 linux/unix 下为/tmp，在 win 下设为读者的目录。

Session.use_Cookies = 1：是否使用 Cookies。

Session.name = PHPSESSID：表示会话 ID。

Session.auto_start = 0：是否自动启用 Session，当为 1 时，在每页中就可以不必调用 Session_start()函数了。

Session.cookie_lifetime = 0：设定 Cookie 送到浏览器后的保存时间，单位为秒。缺省值为 0，表示直到浏览器关闭。

Session.cookie_path = /：Cookies 有效路径。

Session.cookie_domain = Cookies：有效域名。

Session.serialize_handler = php：定义序列化数据的标识，本功能只有 WDDX 模块或 PHP 内部使用，缺省值为 php。

Session.gc_probability = 1：设定每次临时文件开始处理的处理概率。缺省值为 1。

Session.gc_maxlifetime = 1440：设定保存 Session 的临时文件被清除前的存活秒数。

Session.referer_check =：决定参照到客户端的 Session 代码是否要删除。有时出于安全或其他考虑时，会设定不删除。缺省值为 0。

Session.cache_limiter = nocache：设定 Session 缓冲限制

Session.cache_expire = 180：文档有效期，单位为分钟

第三篇

案例实战篇

- 常用模块编程
- 注册登录系统
- 投票系统
- 计数器系统
- 文件管理系统
- 在线图片管理系统
- 功能完善的 BBS 系统
- 电子邮件收发系统
- 在线书店系统

第 12 章　常用模块编程

本章将介绍 PHP 网络编程中最常用的一些模块。通过这些模块的实现，读者能够开阔编程思路，掌握更多的编程技巧。学习这些模块，也能够为开发较大型的系统奠定基础。

12.1　分页显示模块

分页显示模块是数据库管理系统中最常用到的。当从数据库中读取的数据量比较大的时候，如果一次性显示在浏览器中，给用户造成很大的不便，同时也给打印带来了不便。

本节将介绍一个比较通用的分页显示模块。分页显示模块在光盘下的目录为“page”，当用户在浏览器中输入“http://localhost/page/staffList.php”即可分页显示数据库表中的记录。

该模块由两个文件组成，其中“dbConnection.php”用于进行数据库连接的设定，如代码 12-1 所示；而“staffList.php”用于分页显示记录。

【代码 12-1】　dbConnection.php

```
<?php
mysql_connect("localhost", "root", "123456") or die("could not connect");
mysql_select_db("bookstore");  //选择数据库表，用户可以自己选择其他数据库
?>
```

staffList.php 用于显示某个查询语句的记录集，实现了分页，如代码 12-2 所示。

【代码 12-2】　staffList.php

```
<?php
//没页显示记录数
$PageSize = 4;
$StartRow = 0;  //开始显示记录的编号

//获取需要显示的页数，由用户提交
if(empty($_GET['PageNo'])){  //如果为空，则表示第 1 页
   if($StartRow == 0){
      $PageNo = $StartRow + 1;  //设定为 1
   }
}else{
   $PageNo = $_GET['PageNo'];  //获得用户提交的页数
   $StartRow = ($PageNo - 1) * $PageSize;  //获得开始显示的记录编号
}

//因为显示页码的数量是动态变化的
//假如总共有 100 页，则不可能同时显示 100 个链接
//而是根据当前的页数显示一定数量的页面链接
```

```
    //设置显示页码的初始值
    if($PageNo % $PageSize == 0){
       $CounterStart = $PageNo - ($PageSize - 1);
    }else{
       $CounterStart = $PageNo - ($PageNo % $PageSize) + 1;
    }

    //显示页码的最大值
    $CounterEnd = $CounterStart + ($PageSize - 1);
    ?>

    <html>
    <head>
    <title>分页显示记录</title>
    <link rel="stylesheet" href="include/style.css" type="text/css">
    </head>
    <?php
     include 'include/dbConnection.php';
     $TRecord = mysql_query("SELECT * FROM items");
     $result = mysql_query("SELECT image_url,author,price,item_id FROM items
ORDER BY  item_id DESC LIMIT $StartRow,$PageSize");

     //获取总记录数
     $RecordCount = mysql_num_rows($TRecord);

     //获取总页数
     $MaxPage = $RecordCount % $PageSize;
     if($RecordCount % $PageSize == 0){
       $MaxPage = $RecordCount / $PageSize;
     }else{
       $MaxPage = ceil($RecordCount / $PageSize);
     }
    ?>
    <body class="UsePageBg">
    <table width="100%" border="0" class="InternalHeader">
      <tr>
        <td width="24%"><font size=4>分页显示记录</font></td>
        <td width="76%">

           <font size=4><?php print "总共$RecordCount record(s) 条记录  - 当前页:
$PageNo  of $MaxPage" ?></font>
        </td>
      </tr>
    </table>
    <br>
    <table width="100%" border="0" class="NormalTableTwo">
      <tr>
        <td class="InternalHeader">记录序号</td>
        <td class="InternalHeader" >图像路径</td>
        <td class="InternalHeader" >作者</td>
        <td class="InternalHeader" >价格</td>
        <td class="InternalHeader" >图书编号</td>
```

```
  </tr>
<?php
$i = 1;
while ($row = mysql_fetch_array($result, MYSQL_NUM)) {
   $bil = $i + ($PageNo-1)*$PageSize;
?>
  <tr>
    <td class="NormalFieldTwo" ><?php echo $bil ?></td>
    <td class="NormalFieldTwo" ><?php echo $row[0] ?></td>
    <td class="NormalFieldTwo" ><?php echo $row[1] ?></td>
    <td class="NormalFieldTwo" ><?php echo $row[2] ?></td>
    <td class="NormalFieldTwo" ><?php echo $row[3] ?></td>
  </tr>
<?php
  $i++;
}?>
</table><br>
<table width="100%" border="0" class="InternalHeader">
  <tr>
    <td>

      <div align="center">
      <?php
       echo "<font size=4>";
        //显示第一页或者前一页的链接
         //如果当前页不是第一页，则显示第一页和前一页的链接
        if($PageNo != 1){
           $PrevStart = $PageNo - 1;
           print "<a href=staffList.php?PageNo=1>First </a>: ";
           print "<a href=staffList.php?PageNo=$PrevStart>Previous </a>";
        }
         print " [ ";
        $c = 0;


        //打印需要显示的页码
        for($c=$CounterStart;$c<=$CounterEnd;$c++){
           if($c < $MaxPage){
              if($c == $PageNo){
                 if($c % $PageSize == 0){
                    print "$c ";
                 }else{
                    print "$c ,";
                 }
              }elseif($c % $PageSize == 0){
                 echo "<a href=staffList.php?PageNo=$c>$c</a> ";
              }else{
                 echo "<a href=staffList.php?PageNo=$c>$c</a> ,";
              }//END IF
           }else{
              if($PageNo == $MaxPage){
                 print "$c ";
```

```
                break;
            }else{
                echo "<a href=staffList.php?PageNo=$c>$c</a> ";
                break;
            }//END IF
        }//END IF
    }//NEXT

    echo "] ";
    if($PageNo < $MaxPage){  //如果当前页不是最后一页，则显示下一页链接
        $NextPage = $PageNo + 1;
        echo "<a href=staffList.php?PageNo=$NextPage>Next</a>";
    }

    //如果当前页是最后一页，要显示最后一页的链接
    if($PageNo < $MaxPage){
     $LastRec = $RecordCount % $PageSize;
      if($LastRec == 0){
          $LastStartRecord = $RecordCount - $PageSize;
      }
      else{
          $LastStartRecord = $RecordCount - $LastRec;
      }
      print " : ";
      echo "<a href=staffList.php?PageNo=$MaxPage>Last</a>";
      }
       echo "</font>";
    ?>
    </div>
   </td>
  </tr>
</table>
<?php
    mysql_free_result($result);
    mysql_free_result($TRecord);
?>
</body>
</html>
```

分页显示的结果如图 12-1 所示。



图 12-1　分页显示结果

从图 12-1 中可以看出，总共有 6 页，当前为第 4 页，如果单击“Next”按钮，则会进入到第 5 页，此时显示的页面导航数量变成了两个，如图 12-2 所示。



图 12-2 显示下一页

至此一个功能比较完善的分页显示模块介绍完毕。用户只需要修改其中的数据库名称、数据库表名称，以及对应的字段信息，即可改成符合自己需要的分页显示系统。

在分页显示过程中，可以设置分页显示的记录数量，应该根据实际需要来设置。另外在显示分页的过程中，为了防止页数太多，不能一次性将所有的页面链接显示出来，程序中可以根据每页显示的记录数量来动态显示分页的链接数量，如每页显示的记录数为 4 条，则每页显示的页面导航数量页为 4 条。如果到最后，只剩下两页，则显示剩余的两个页面的链接，如图 12-2 中所示的一样。

12.2 在线数据编辑模块

本章将介绍一个在线数据编辑模块，该模块类似其他编程语言中的 DataGrid 数据控件。在浏览器中显示一个表格用来显示记录，同时用户可以方便的编辑、删除以及添加新记录，而不需要通过额外的提交按钮。

该模块的目录名称为“Datagrid”，当用户在浏览器中输入“http://localhost/Datagrid/demo1.php”，即可以表格形式显示记录，如图 12-3 所示。

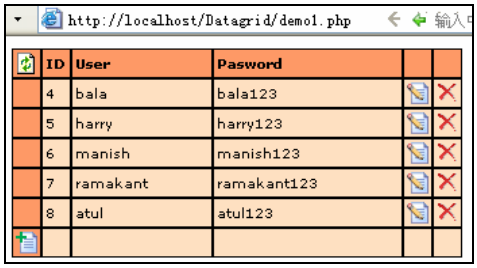


图 12-3 在线数据编辑模块

在图 12-3 中，可以单击按钮删除一条记录、单击编辑一条记录、单击新增一条记录，以及单击取消数据的编辑。

编辑记录的界面如图 12-4 所示。

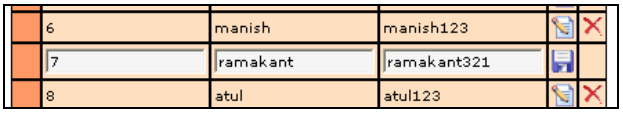


图 12-4 编辑记录

记录编辑后，单击可以保存编辑后的数据，如图 12-5 所示。

从图 12-5 中可以看出，编号为 7 的记录，其中“password”列的值由“ramakant123”被

改成了“ramakant321”。

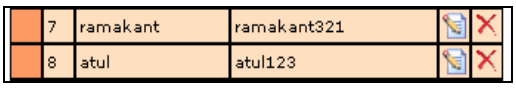


图 12-5　编辑成功

添加新记录的界面如图 12-6 所示。



图 12-6　添加新记录

本模块的核心就是显示一个框架式的界面，以及实现在线编辑。另外，数据库编辑同其他章介绍的一样，也是使用基本的查询、删除、插入和编辑等 SQL 语句来实现。

本模块由以下几个文件组成：

（1）demo1.php，用于演示在线编辑功能的实现；

（2）delete.php，用于删除记录；

（3）save.php，用于保存修改、新增的记录；

（4）update.php，用于编辑记录；

（5）drawDataGrid.class.php，用于创建一个 drawDataGrid 类，该类显示一个框架式界面，可以在线编辑和添加记录。

由于数据库操作，其他章节有详细介绍，下面重点介绍 drawDataGrid 类的实现，该类是用于显示表格的，可以调用事先设定好的表格方法和属性输出表格。

实现该类的文件为 drawDataGrid.class.php，其代码如代码 12-3 所示。

【代码 12-3】　drawDataGrid.class.php

```
    <?php
    class drawDataGrid
    {
    var $headerColor="#FF9966";
    var $cellColor="#FFE0C1";
    var $borderSize=1;
        function TableStart($tblWidth)  //输出表格关键字 table 及属性设置
       {
            echo "<table id='grdTbl' width='".$tblWidth."%' cellspacing='0'
cellpadding='2' border='".$this->borderSize."' bordercolor='#000000' class=
'gridtbl'>";
            echo chr(13);
        }
        function TableEnd()  //输出表格结束符
       {
            echo "</table>";
            echo chr(13);
       }
        function RowStart($idVal)  //输出表格行开始
       {
            echo "<tr id='".$idVal."'>";
```

```
            echo chr(13);
        }
        function RowEnd()  //输出表格行结束
        {
            echo "</tr>";
            echo chr(13);
        }
        function mainCellStart($cellWidth)  //表格标题单元开始，也就是第一行的表格单元
        {
            echo "<td class='tdInset' bgcolor='".$this->headerColor."' width=
'".$cellWidth."%'>";
            echo chr(13);
        }
        function mainCellEnd()  //表格标题单元单元结束
        {
            echo "</td>";
            echo chr(13);
        }
        function CellStart()  //其他表格单元开始
        {
            echo "<td bgcolor='".$this->cellColor."' class='tdInset'>";
            echo chr(13);
        }
        function CellEnd()  //表格单元结束
        {
            echo "</td>";
            echo chr(13);
        }
    }

    ?>
```

而调用 drawDataGrid 类是在 demo1.php 文件中实现的，如代码 12-4 所示。

【代码 12-4】 demo1.php

```
    <?php
    //包含类文件
    include("drawDataGrid.class.php");

    if (isset($_REQUEST['editTbl']))  //判断是否为编辑记录
    {
        $editTbl = $_REQUEST['editTbl'];
    }
    else
    {
        $editTbl=0;
    }
    if (isset($_REQUEST['newTbl']))  //判断是否为插入新记录
    {
        $newTbl = $_REQUEST['newTbl'];
    }
    else
```

```
{
    $newTbl=0;
}
if (isset($_REQUEST['rowNo']))  //行编号
{
    $rowNo = $_REQUEST['rowNo'];
}
else
{
    $rowNo=0;
}
if (isset($_REQUEST['curpage'])) //获取当前页
  $curpage = $_REQUEST['curpage'];
else
  $curpage=0;

if ($curpage == 0)  // 如果当前页为0 ,则设置为1
{
    $curpage = 1;
}
$db_conn=mysql_connect("localhost","root","123456");  //连接数据库
mysql_select_db("php",$db_conn);  //选择数据库php，需要在实例运行之前创建
$query = "select * from logins";  //从logins表中查询记录
$result=mysql_query($query,$db_conn);  //执行查询
$num_rows = mysql_num_rows($result);  //获取所有记录数量
$rowLower = 0;
$rowUpper = $num_rows;
$colLower = 0;
//获取字段信息
$fields = mysql_list_fields("php", "logins", $db_conn);
//获取字段数量
$columns = mysql_num_fields($fields);
$colUpper = $columns + 2 ;
$updateImg = true;  //编辑图像显示
$deleteImg = false;
?>
<HTML>
<HEAD>
<title>PHP DATAGRID - by Bala</title>
<META NAME="GENERATOR" Content="Microsoft FrontPage 4.0">
<link rel="stylesheet" href="stylesheet.css" type="text/css">
<script>
function cancel()  //取消记录编辑、插入
    {
    document.tmpFrm.rowNo.value = 0;
    document.tmpFrm.editTbl.value = 0;
    document.tmpFrm.newTbl.value = 0;
    document.tmpFrm.submit();
    }//end of check
function check(i)  //  显示编辑记录界面
    {
    document.tmpFrm.rowNo.value = i;
```

```
    document.tmpFrm.newTbl.value = 0;
    document.tmpFrm.submit();
    }//end of check
function checkNew(i)   //显示插入新记录界面
    {
    document.tmpFrm.rowNo.value = 0;
    document.tmpFrm.editTbl.value = 0;
    document.tmpFrm.submit();
    }//end of check
function deleteYesNo(i)  //显示确认删除对话框
    {
    var ok;
    ok = confirm("Are you sure you want to delete this record?");
    if(ok)
        {
        document.dgridFrm.del.value = i;
        //val.value = i;
        //alert(val.value);
        document.dgridFrm.action = "delete.php";
        document.dgridFrm.submit();  //提交记录删除
        }//end if
    }//end of deleteYesNo()
function newRecord()  //提交新记录
    {
        if(document.dgridFrm.item(1).value == "")
            {
                alert("Please enter User Name!")
                document.dgridFrm.item(1).focus();
            }//end if
        else
            {
                if(document.dgridFrm.item(2).value == "")
                    {
                        alert("Please enter Password!")
                        document.dgridFrm.item(2).focus();
                    }//end if
                else
                    {
                        document.dgridFrm.action = "save.php";
                        document.dgridFrm.submit();
                    }//end else
            }//end else

    }//end of deleteYesNo()
</script>
</HEAD>
<BODY>
<form name="tmpFrm" method="post">
    <input type="hidden" name="editTbl" value="1">
    <input type="hidden" name="newTbl" value="1">
    <input type="hidden" name="rowNo">
</form>
```

```
    <form name="dgridFrm" method="post" action="update.php">
    <input type="hidden" name="del">
    <table width="60%" border="0">
    <?
    if (isset($_POST['mess']))
    {
    ?>
    <tr>
        <td align="center">
        <?
       echo "<font color=\"red\" face=\"verdana\">".$_POST['mess']. "</font><br>";
       ?>
        </td>
    </tr>
    <tr>
        <td> </td>
    </tr>
    <?
    }
    ?>
    <tr>
        <td align="center">
    <?
    $tblWidth = 100;
    $tblBorder = 0;
    $title = array(1=>"ID",2=>"Password",3=>"Email");
    $dgrid = new drawDataGrid;  //创建类实例
    $dgrid->TableStart($tblWidth);
    $dgrid->RowStart(0);
    for ($i=0; $i <= mysql_num_fields($result); $i++) //显示表头
    {
       if ($i ==0)
       {
         $dgrid->mainCellStart(5);
         echo "<a href='JavaScript:cancel();'><img src='recycle.gif' border='0'
alt='Cancel'></a>";
         $dgrid->mainCellEnd();
       }
       else
       {
        if (($i ==2)||($i ==3))  //显示字段名称
         {
         $dgrid->mainCellStart(0);
         echo "<b>".ucfirst(mysql_field_name($result, $i-1))."</b>";
         $dgrid->mainCellEnd();
         }
         else
         {
          $dgrid->mainCellStart(5);
           echo "<b>".ucfirst(mysql_field_name($result, $i-1))."</b>";
          $dgrid->mainCellEnd();
         }
```

```
       }
    }
    $dgrid->mainCellStart(5);
    echo " ";
    $dgrid->mainCellEnd();
    $dgrid->mainCellStart(5);
    echo " ";
    $dgrid->mainCellEnd();
    $dgrid->RowEnd();
        //显示表格的具体内容
       for ($r = $rowLower; $r < $rowUpper; $r++)
       {
         $row = mysql_fetch_array($result);
         $dgrid->RowStart($r);
         for ($c = $colLower; $c <= $colUpper; $c++)
         {
           if ($c ==$colLower)
        {

             $dgrid->mainCellStart(0);
          echo " ";
              $dgrid->mainCellEnd();
           }
        else
           {
             if ($row[0]==$rowNo)
             {
               if ($c <= $columns)
               {
               $dgrid->CellStart();
                 echo "<input type='text' name='".mysql_field_name($result, $c-1).
"' taborder='0' value='".trim($row[$c - 1])."'>";  //显示文本框
               $dgrid->CellEnd();
               }
               else if ($c==$columns + 1)
               {
                $dgrid->CellStart();
                echo "<a href='JavaScript:document.dgridFrm.submit();'><img
src='save.gif' border='0' alt='Save'></a>";  //显示保存图像
                $dgrid->CellEnd();
               }
               else
               {
                $dgrid->CellStart();
                echo " ";
                $dgrid->CellEnd();
               }
             }
             else
             if ($c <= $columns)
              {
             $dgrid->CellStart();
```

```
          echo trim($row[$c - 1]);
          $dgrid->CellEnd();
             }
             else
              if ($c==$columns + 1)
              {
               $dgrid->CellStart();
               echo "<a href='JavaScript:check(".$row[$c - 4].
            ");'><img src='edit.gif' border='0' alt='Edit'></a>";  //显示编辑图像
            $dgrid->CellEnd();
              }
              else
              {
               $dgrid->CellStart();
            echo "<a href='JavaScript:deleteYesNo(".$row[$c - 5].");'><img src=
'delete.gif' border='0' alt='Delete'></a>";  //显示删除图像

            $dgrid->CellEnd();
              }
           }
          }
         $dgrid->RowEnd();
    }
    if ($newTbl == 0)
    {
       $dgrid->RowStart($num_rows);
       $dgrid->mainCellStart(0);
       echo "<a href='JavaScript:checkNew(".($r + 1).");'><img src='add.gif'
border='0' alt='New Record'></a>";  //显示添加新记录图像
       $dgrid->mainCellEnd();
    //   echo $colLower." ".$colUpper;
       for ($c = $colLower; $c < $colUpper; $c++)
       {
       $dgrid->CellStart();
       if ($c==$colLower)
          echo " ";
       else
          echo " ";
       $dgrid->CellEnd();
       }
       $dgrid->RowEnd();
    }
    if ($newTbl == 1)
    {
        $dgrid->RowStart($num_rows + 1);
        $dgrid->mainCellStart(0);
        echo " ";
        $dgrid->mainCellEnd();
       for ($c = $colLower; $c < $colUpper; $c++)
       {

              if ($c==0)
```

```
                {
                 $dgrid->CellStart();
                 mysql_data_seek($result, $num_rows-1);
                // $row = mysql_fetch_array($result);
                 $row = mysql_fetch_row($result);
                 echo $row[0]+1;
                 $dgrid->CellEnd();
                }
                else if (($c) < $columns)
                {
                $dgrid->CellStart();
                 echo "<input type='text' name='".mysql_field_name($result, $c-1).
"' taborder='0'>";  //显示文本框
                $dgrid->CellEnd();
                }
                else if ($c==$columns)
                {
                 $dgrid->CellStart();
                 echo "<a href='JavaScript:newRecord();'><img src='save.gif' border
='0' alt='Save'></a>";  //显示保存图像
                 $dgrid->CellEnd();
                }
                else
                {
                 $dgrid->CellStart();
                 echo " ";
                 $dgrid->CellEnd();
                }
         }
        $dgrid->RowEnd();
       }
    $dgrid->TableEnd();
    ?>
    </td>
    </tr>
    </table>
    </form>
    </BODY>
    </HTML>
```

本模块的其他数据库操作部分，由于篇幅限制，这里就不介绍了。本实例代码比较容易理解，主要根据用户的选择，显示对应的界面。读者应重点学习类的运用，以及各种界面的处理技术，包括如何通过 JavaScript 来提交数据和通过 JavaScript 来控制表单属性。

12.3 文件上载模块

文件上载模块也是一个常用的模块。在本书后面的“文件管理系统”和“在线图片管理系统”中，都使用到了文件上载模块。

本实例的目录为“upload”，只有一个文件，文件名为 upload.php。用户在浏览器中输入“http://localhost/upload/upload.php”即可进入文件上载界面，如图 12-7 所示。

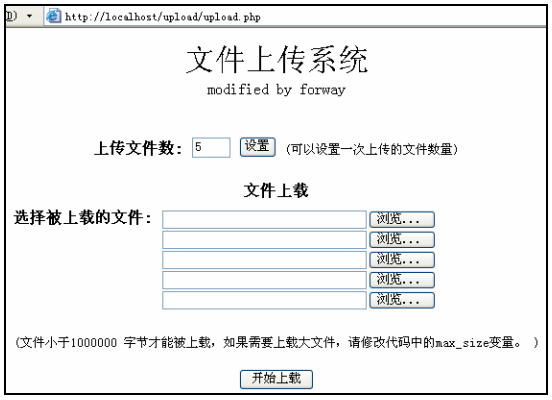


图 12-7　文件上载界面

此处可以设置一次上传的文件数，比如将上传文件数改成 2，则会显示两个文件选择文本框，如图 12-8 所示。

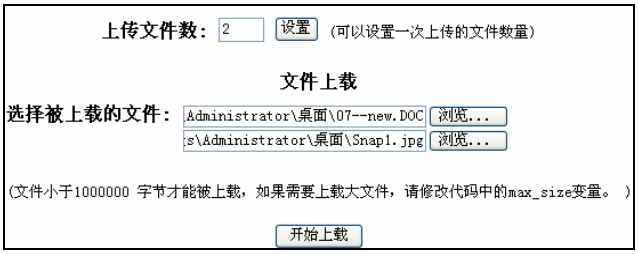


图 12-8　修改文件上载数量

文件上载成功后的界面如图 12-9 所示。

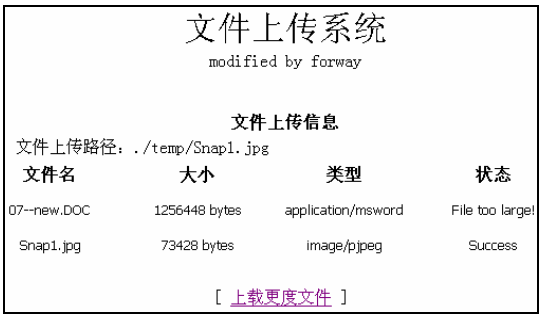


图 12-9　文件上载成功

从图 12-9 中可以看出，snap1.jpg 已经被上载成功，由于 07-new.doc 文件大小超过了事先设定的最大值，因此上载失败。

upload.php 文件的代码如代码 12-5 所示。

【代码 12-5】　upload.php

```
<?php
  if (isset($_POST['set'])) {
    $upload_slots = $_POST['slots'];
  } else {
```

```
      $upload_slots = 5; // <---------- 默认文件上传数量
     }
     $max_size = 1000000; // 文件最大上传字节数
     $location = "./temp/";    // 文件被上传后的目录
   ?>
   <html>
    <center>
    <p><font size=6 face=tahoma>文件上传系统</font><br>modified by forway <br>
</p><br>
     <?php if (! isset($_POST['upload'])){ ?>
     <form action="upload.php" method="POST">
     <table width="635">
       <tr>
         <td width="100%">
           <p align="center"><b>上传文件数:
             <input name="slots" type="text" id="slots3" value="<?php echo $upload
_slots; ?>" size="4" maxlength="2">
             </b>
             <input name="set" type="submit" id="set" value="设置">
             <font size="2">(可以设置一次上传的文件数量) </font>
         </table>
     </form>
     <form method="POST" enctype="multipart/form-data" action="upload.php">
     <table width="635">
       <td width="100%">
          <p align="center"><b>文件上载</b></td>
       </tr>
       <tr>
         <td width="100%">
           <table border="0" width="100%" cellpadding="4">
             <tr>
               <td width="30%" align="right" valign="top"><b>选择被上载的文件:</b></td>
               <td width="70%">
                 <input type="hidden" name="MAX_FILE_SIZE" size="5200000">
   <?php
     //动态输出文件上传表单控件
     for($count = 1; $count < $upload_slots+1; $count++) {
       echo '<input type="file" name="upload'.$count.'" size="29"><br>';
     }
   ?>
               </td>
             </tr>
           </table>
         </td>
       </tr>
     </table>
     <br>
     <font size="2">(文件小于<?php echo $max_size; ?> 字节才能被上载，如果需要上载大
文件，请修改代码中的 max_size 变量。
     )</div> </div> </font>
     <p align="center">
       <input type="hidden" name="slots" value="<?php echo $upload_slots; ?>">
```

```
      <input type="submit" value="开始上载" name="upload">
    </p>
    </form>
    <? } else { ?>
    <div align="center">
    <center>
      <table width="674">
        <tr>
          <td width="100%">
            <p align="center"><b>文件上传信息</b></td>
        </tr>
        <tr>
          <td width="100%">
            <table border="0" width="100%" cellspacing="3" cellpadding="6">
              <tr>
                <td width="25%" align="center"><b>文件名</b></td>
                <td width="25%" align="center"><b>大小</b></td>
                <td width="25%" align="center"><b>类型</b></td>
                <td width="25%" align="center"><b>状态</b></td>
              </tr>
  <?php
    // 循环检查每个提交的文件
    for ($num = 1; $num < $_POST['slots']+1; $num++){
      $event = "Success";
      // 检查是否有文件上传
      if (! $_FILES['upload'.$num]['name'] == ""){
        if ($_FILES['upload'.$num]['size'] < $max_size) {
            echo "文件上传路径: ".$location.$_FILES['upload'.$num]['name'];
          move_uploaded_file($_FILES['upload'.$num]['tmp_name'],$location.$_
FILES['upload'.$num]['name']) or $event = "Failure";
          } else {
           $event = "File too large!";
          }
          // 显示上载文件的信息
         echo "<tr>";
          echo "  <td width='25%' align='center'><font face='Tahoma' size='2'>
".$_FILES['upload'.$num]['name']."</td>";
          echo "  <td width='25%' align='center'><font face='Tahoma' size='2'>".
$_FILES['upload'.$num]['size']." bytes</td>";
          echo "  <td width='25%' align='center'><font face='Tahoma' size='2'>
".$_FILES['upload'.$num]['type']."</td>";
          echo "  <td width='25%' align='center'><font face='Tahoma' size='2'>".
$event."</td>";
          echo "</tr>";
       }
    }
  ?>
            </table>
          </td>
        </tr>
      </table>
      <p>[ <a href="upload.php">上载更度文件</a> ] </p>
```

```
    </center>
   </div>
  <? } ?>
  </html>
```

可以修改每个上载文件的最大字节数，直接修改代码中的 max_size 变量即可。另外 $location 变量用于决定文件的上载路径。本系统默认路径是当前目录下的“temp”子目录。

12.4 聊天室模块

在早期的综合性网站中，聊天室是一个不可缺少的模块。随着网络技术的发展，一些即时通信工具逐步诞生，如 QQ、MSN 等，因此这些基于 Web 的在线聊天室逐渐失去了原来的辉煌。但是现在有了比较专业的聊天室，能够通过插件等各种技术提供文本、视频、音频等聊天方式。

本模块只是一个基于文本的简单聊天室，目的是起到一个抛砖引玉的作用。

聊天室由以下文件组成：

（1）index.php，用于显示登录界面；

（2）login.php，用于显示登录是否成功，如果登录成功，则显示进入聊天室按钮，同时也显示退出聊天室按钮；

（3）frames.php，该文件为框架文件，由 outputchat.php、outputusers.php 以及 input.php3 个文件组成；

（4）outputchat.php，用于显示聊天信息；

（5）outputusers.php，用于显示在线用户；

（6）input.php，用于让用户输入聊天信息；

（7）logout.php，用于显示退出聊天室的信息；

（8）chat.txt，用于保存聊天信息；

（9）users.txt，用于保存在线用户列表。

12.4.1 用户登录界面（index.php）

index.php 文件用于显示用户登录界面。聊天室系统的目录名称为“chat”，输入“http://localhost/chat/index.php”，即可进入聊天室登录页面，如图 12-10 所示。



图 12-10 聊天室登录界面

【代码 12-6】 index.php

```
<html>
<head>
```

```
    <?php
    $readdata = file("hits.txt");  //从 hits.txt 文件中读取访问的人数
    $readdata[0] = $readdata[0] + 1;  //将人数添加 1
    $writedata = fopen("hits.txt", "w");
    fputs ($writedata, $readdata[0]);  //写回到文件
    fclose ($writedata );
    ?>
    <script language="JavaScript" type="text/JavaScript">
    function checknickname() {  //用 javascript 检查用户名是否为空
        if (document.login.nickname.value.length < 1) {
            alert ("昵称不能为空!");
            return (false);
            }
        return (true);
    }
    </script>
    <title> | forway 聊天室 | </title>
    <link href="styles.css" rel="stylesheet" type="text/css">
    <body>
    ……省略部分 html 代码
                                    <td>昵称:</td>
                                    <td><input name="nickname" type="text" id=
"nickname" size="30"></td>
                            </tr>
                            <tr>
                                    <td valign="top">选择文字颜色:</td>
                                    <td>
                                        <select name="color" id="color">
                                            <option value="00FF00">Green</option>
                                            <option value="3399FF">Blue</option>
                                            <option value="FFFF00">Yellow</option>
                                            <option value="FF3300">Red</option>
                                            <option value="FFFFFF">White</option>
                                        </select>
                                    </td>
                            </tr>
                            <tr>
                                    <td colspan="2" align="center" valign="top">
                                        <br><input name="Submit" type="submit" class=
"button" value="登录!">
    ……省略部分 html 代码
```

该文件主要显示登录表单界面，然后将访问人数添加 1，并保存到 hits.txt 文件中。

12.4.2　登录成功界面（login.php）

登录成功后进入 login.php 页面，如图 12-11 所示。



图 12-11　登录成功

该界面主要显示两个功能按钮，用于进入具体的聊天室界面或者退出登录。

【代码 12-7】 login.php

```
    <html>
    <head>
    <script language="JavaScript" type="text/JavaScript">
    var chatwindow
    function openchat() {  //该函数打开聊天对话框，frames.php 为聊天的框架文件
    <?php
        echo "chatwindow = window.open(";
        echo "'frames.php?nickname=" . $_POST["nickname"] . "&color=" . $_POST
["color"] . "'";
        echo",'chat','toolbar=no,location=no,status=no,menubar=no, scrollbars=
no,resizable=no,width=700,height=700');";  //设置浏览器的显示属性
    ?>
    }
    function logout() {  //该函数弹出注销对话框
    <?php
        echo "window.open(";
        echo "'logout.php?nickname=" . $_POST["nickname"];
        echo"','logout','toolbar=no,location=no,status=no,menubar=no,scrollb-
ars= no,resizable=no,width=200,height=200');";
    ?>
    parent.location = "exit.htm"
    chatwindow.close();
    }
    </script>
    <title> | MataChat |
    <?php
        global $newline;
        $newline = chr(13) . chr(10);  //回车换行符
        global $userok;
        $userok = 1;
        $dataread = file("users.txt"); //读取用户列表，users.txt 文件保存在线聊天用户
        for ($i = 0; $i <= count($dataread); $i++) {  //判断该用户是否已经登录
            if ($dataread[$i] == $_POST["nickname"] . $newline)  {
                $userok = 0;  //如果已经登录，则表示不能再次登录，设 userok 为 0
            }
        }
        if ($userok) {  //如果 userok 不为 0，表示之前没有登录
            echo "Logged In";
        }
        else {  //如果为 0，表示之前已经登录，不能再次登录
            echo "Login failed";
        }
    ?>
    | </title>
    <link href="styles.css" rel="stylesheet" type="text/css">
    </head>
    <body onUnload=  //窗体下载事件
    <?php
```

```
        echo "'";
        if ($userok) {
            echo "javascript:logout();";  //退出登录
        }
        echo "'";
    ?>
    >
    <table width="100%" height="100%"  border="0" cellpadding="0" cellspacing="0">
        <tr>
        <td align="center" valign="middle">
    <?php
        if ($userok) { //如果之前没有登录
            $appenddata = fopen("users.txt", "a");  //添加用户名到 users.txt 文件
            fputs ($appenddata, $_POST["nickname"] . $newline);
            fclose ($appenddata);
            echo "你已经登录!<br><br><input type='button' class='button' onClick=
'javascript:openchat();' value='进入聊天室!'>";
            echo "<br><br><input type='button' class='button' onClick='javascript:
logout();' value='退出!'";
        }
        else {  //如果没有登录成功，输出提示信息
            echo "Invalid Nickname!<br><br><input type='button' class='button'
onClick='javascript:history.back()' value='Back!'>";
        }
    ?>
            </td>
        </tr>
    </table>
    </body>
    </html>
```

12.4.3　聊天室的主界面（frames.php）

frames.php 文件为聊天室的框架文件，在 login.php 文件的界面上单击“进入聊天室”即可进入聊天室，如图 12-12 所示。

【代码 12-8】　frames.php

```
    <html>
    <head>
    <title>
    <?php
        echo " | forway 聊天室 | " . $_GET["nickname"] . " | ";
    ?>
    </title>
    </head>
    <frameset rows="500,75,*" frameborder="NO" border="0" framespacing="0">
        <frame src="outputchat.php" name="topFrame" scrolling="yes" noresize>
        <frame src="outputusers.php" name="topFrame" scrolling="yes" noresize>
        <frame src=
    <?php
        echo " 'input.php?nickname=" . $_GET["nickname"] . "&color=" . $_GET
```

```
["color"] . " ' ";
    ?>
        name="mainFrame">
    </frameset>
    </html>
```

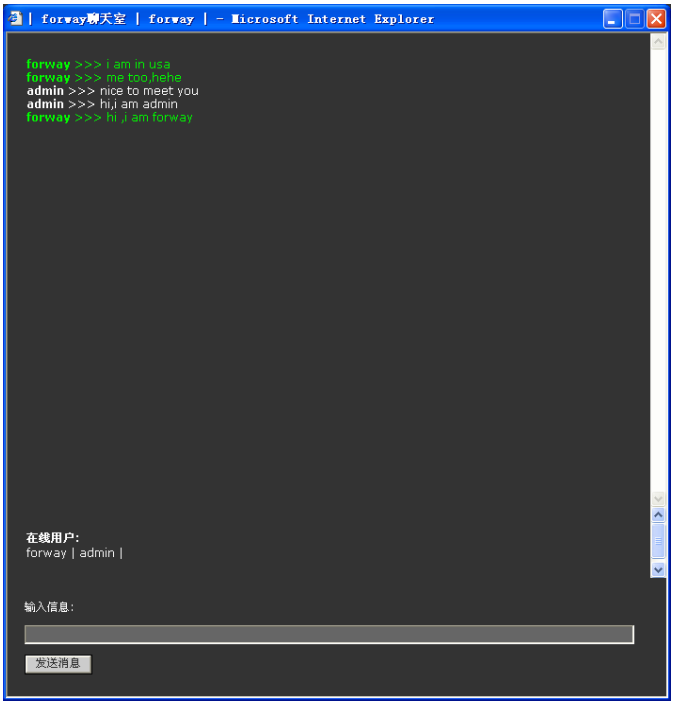


图 12-12　聊天室主界面

框架文件由 3 个文件组成，其中 outputchat.php 用于显示当前聊天的内容、outputusers.php 用于显示当前在线聊天用户，input.php 用于输入聊天内容并发送。

12.4.4　显示聊天内容界面（outputchat.php）

outputchat.php 文件用于输出每个聊天客户的聊天内容。聊天室将聊天信息临时保存在 chat.txt 文件中。显示聊天内容的时候只需要直接读取该文件的内容即可。

【代码 12-9】　outputchat.php

```
    ……省略部分内容
    <meta http-equiv="refresh" content="3">
    </head>
    <body>
    <table width="100%"  border="0" cellspacing="0" cellpadding="10">
        <tr>
        <td>
    <?php
        $readdata = file("chat.txt");  //读取 chat.txt 文件内容
        for ($i=0; $i <= count($readdata); $i++) {
            echo $readdata[$i] . "<br>";  //显示所有聊天信息
        }
```

```
    ?>
    ……省略部分内容
```

为了保证聊天信息的及时更新，页面是自动刷新的，自动刷新由语句“<meta http-equiv="refresh" content="3">”实现，其中“content="3"”表示每隔 3s 刷新一次，而“http-equiv="refresh"”表示自动刷新。

12.4.5　显示在线用户列表界面（outputusers.php）

outputusers.php 文件用于显示当前在线的聊天用户列表。当前在线用户列表保存在 users.txt 文件中，显示列表的时候只需要从该文件读取数据并显示出来。

【代码 12-10】　outputusers.php

```
    ……省略部分内容
            <b>在线用户:</b><br>
    <?php
    $readdata = file ("users.txt");  //读取用户信息
    for ($i=0; $i <= count($readdata) - 1; $i++) {
        if ($readdata[$i] != "") {
            echo $readdata[$i] . "| ";  //循环显示所有在线用户
        }
    }
    ?>
    ……省略部分内容
```

12.4.6　输入聊天内容界面（input.php）

input.php 文件用于显示发送信息的表单控件，让用户输入聊天信息，并发送。

【代码 12-11】　input.php

```
    <html>
    <head>
    <?php
        //本聊天室支持自定义表情，如当用户输入:)时，会自动显示一个笑脸
        function replaceHTML($message) {  //替换聊天内容中的部分信息
            $replace = array(
                "<" => "&lt;",
                ">" => "&gt;",
                ":)" => "<img src='smilies/smile.gif' border='0'>",
                ":-)" => "<img src='smilies/smile.gif' border='0'>",
                ":D" => "<img src='smilies/smile.gif' border='0'>",
            );
           $message = str_replace(array_keys($replace), array_values($replace),
$message);
            return $message;
        }
    ?>
    <script language="JavaScript" type="text/JavaScript">
    function setfocus() {
        document.senddata.message.focus();  //文本框获得焦点
    }
```

```
    </script>
    <title>Test</title>
    <link href="styles.css" rel="stylesheet" type="text/css">
    <style type="text/css">
    input {
    <?php
        echo "color: #" . $_GET["color"] . ";";  //设定字体颜色，用户登录时选择设定
    ?>
    }
    </style>
    </head>
    <body onLoad="javascript:setfocus();">
     <form name="senddata" method="Get" action="input.php">
     <?php
     echo "<input name='nickname' type='hidden' id='nickname' value=" . "'" .
$_GET["nickname"] . "'>";  //用一个隐含的控件保存用户昵称
     echo "<input name='color' type='hidden' id='color'  value=" . "'" . $_GET
["color"] . "'>";  //用一个隐含的控件保存字体颜色
     ?>
    ……省略部分内容
        if (!file_exists("chat.txt")) {  //如果 chat.txt 文件不存在，则退出
            exit();
        }
        else {
            $newline = chr(13) . chr(10);
                if ($_GET["message"] != "") {  //如果用户聊天内容不为空
                    //将一些表情字符替换成实际的图像路径
                    $replacedmessage = replaceHTML($_GET["message"]);
                    $output = "<font color='#" . $_GET["color"] . "'><b>" . $_GET
["nickname"] . " &gt;&gt;&gt; </b>" .  $replacedmessage . "</font>" .  $newline;
                    $dataread = file("chat.txt");  //打开文件
                        for ($i=0; $i <= count($dataread); $i++) {
                            $output .= $dataread[$i];  //将新发表的内容放在文件开头
                        }
                    $datawrite = fopen("chat.txt", "w");
                    fputs ($datawrite, $output);  //将更新后的内容写入文件
                    fclose ($datawrite);
                }
        }
    ?>
    ……省略部分内容
```

以上介绍了一个聊天室的基本实现，该聊天室具备了各种常用功能，当然，读者可以将其扩展，实现更多的功能。

第 13 章　注册登录系统

注册登录系统是任何一个综合系统都必须具备的。通过注册、登录模块，网络管理员才能够更好地管理整个网站。

本章将介绍一个功能比较完善的注册、登录模块，将实现以下功能：

（1）用户注册；

（2）用户登录；

（3）发送注册码，邮寄给注册客户；

（4）通过注册码激活账号；

（5）对登录密码进行加密；

（6）产生登录验证码。

13.1　系统整体设计

根据注册登录系统的功能需求，本系统将由以下文件组成：

（1）config.php，数据库基本设置文件；

（2）install.php，系统自动安装文件；

（3）login.php，登录文件；

（4）signup.php，注册文件；

（5）forgot.php，忘记密码时，找回密码文件；

（6）activate.php，账号激活文件；

（7）add.php，将注册数据保存到数据库中；

（8）logoff.php，注销登录。

13.2　数据库安装配置

系统的注册、登录基本上都是通过数据库进行管理的。因此首先要创建数据库，数据库名称为“members”。然后创建数据库表，用户注册登录使用的数据库表主要有一个，表名为“als_signup”，该表的 SQL 语句如下所示：

```
CREATE TABLE als_signup(
username VARCHAR(20),   //用户名
password VARCHAR(20),    //用户密码
mailadres VARCHAR(100),   //Email 地址
actnum VARCHAR(20),   //激活码
userlevel TINYINT,    //用户权限
signupdate VARCHAR(16),   //是否激活
lastlogin VARCHAR(16),   //最近登录成功时间
```

```
    lastloginfail BIGINT,   //最近登录失败时间
    numloginfail TINYINT   //登录失败次数
    )
```



在进行用户注册的时候，可以根据需要输入更多的信息。本书后面的 BBS 系统、在线书店系统，都有比较详细的介绍，读者可以对照学习。

本书提供了全自动的数据库安装文件，用户只需要在浏览器中输入“http://localhost/register/install.php”，即可自动完成数据库创建和数据库表的创建。

在执行该文件之前，首先要打开 config.php 文件，修改其中的数据库连接用户名和密码。config.php 文件如代码 13-1 所示。

【代码 13-1】 config.php

```
    <?php
    $server = "localhost"; // often localhost
    $username = "root";             // 用户名
    $password = "123456";           // 密码
    $database = "";             // 要创建的数据库名称，如果不设置，则默认为 'members'
    $DBprefix = "ALS";          // 创建表的前缀
    //省略部分代码
    // if the prefix is not empty add an _ (underscore)
    if ($DBprefix != ""){$DBprefix = $DBprefix . "_";}
    // 判断数据库是否已经创建
    if ($database == ""){
        // 调用 use 语句
        $query = "use members";
        //如果成功，则表示数据库存在
        // 如果返回 null，表示不存在
        if (mysql_query($query) == null){
            //如果不存在则需要创建
            $query = "CREATE DATABASE members";
            $result = mysql_query($query);
            if ($result == 1){
            $conn = mysql_connect($server,$username,$password) or die ($could
NotConnectMysql);
            //选择数据库
            mysql_select_db($database,$conn) or die ($couldNotOpenDatabase);
            }else{echo "Error while creating database (Errornumber ". mysql_
errno() .": \"". mysql_error() ."\")<br>";}
        }
        else{
            // 如果数据库已经存在，则直接连接
            $database= "members";
            $conn = mysql_connect($server,$username,$password) or die ($could
NotConnectMysql);
            mysql_select_db($database,$conn) or die ($couldNotOpenDatabase);
            //echo "数据库选择成功了!";
        }
    }
    else{
        // 连接数据库
```

```
        $conn = mysql_connect($server,$username,$password) or die ($couldNot
ConnectMysql);
        mysql_select_db($database,$conn) or die ($couldNotOpenDatabase);
    }
    ?>
```

安装程序 install.php 需要包含 config.php 文件，install.php 文件如代码 13-2 所示。

【代码 13-2】　install.php

```
    <?php
    //省略部分代码
    include("config.php");
    // make the signup table
    $query = "CREATE TABLE ".$DBprefix."signup(username VARCHAR(20), password
VARCHAR(20), mailadres VARCHAR(100), actnum VARCHAR(20), userlevel TINYINT,
signupdate VARCHAR(16), lastlogin VARCHAR(16), lastloginfail BIGINT,numloginfail
TINYINT)";
    $result = mysql_query($query);
    // 检查是否创建成功
    if ($result == 1){echo "signup Table succesfully created.<br>";}
    //如果失败，输出提示信息
    else{echo "Error while creating signup table (Errornumber ". mysql_errno() .":
\"". mysql_error() ."\")<br>";}
    //省略部分代码
    ?>
```

install.php 主要是调用 CREATE TABLE 关键字创建一个数据库表，表名为 signup。

13.3　系统的实现

接下来介绍整个注册登录系统的实现。

13.3.1　系统注册模块

要进入注册模块，可以从登录模块中的链接进入，也可以直接在浏览器中输入“http://localhost/register/signup.php”，进入注册模块。注册模块的显示界面如图 13-1 所示。



图 13-1　注册页面

新用户注册是由文件 signup.php 实现的，如代码 13-3 所示。

【代码 13-3】　signup.php

```
    <html>
    <head>
```

```
    <title>Sign-up</title>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
    </head>
    <body>
    <form name="form1" method="post" action="add.php">
      <h1>新用户注册</h1>
      <p> 用户名:
        <input name="username" type="text" id="Username">
        <br>
        密  码:
        <input name="password" type="password" id="password">
        确认密码:
        <input name="confirmpassword" type="password" id="confirmpassword">
        <br>
        Email:
        <input name="mailadres" type="text" id="mailadres">
        <br>
        <input type="submit" value="注册">
        <br>
        如果已经有账号，可以单击 <a href="login.php">here</a>登录.</p>
    </form>
    </body>
    </html>
```

signup.php 文件主要显示注册表单，而将数据写入到数据库是由文件 add.php 来实现的，如代码 13-4 所示。

【代码 13-4】 add.php

```
    <?PHP
    // 获取用户提交数据
    $username1 = $_POST["username"];
    $mailadres1 = $_POST["mailadres"];
    $password1 = $_POST["password"];
    $confirmpassword1 = $_POST["confirmpassword"];
    // 保存激活码的变量
    $actnum = "";
    //包含设置文件
    include("config.php");
    //如果不需要通过电子邮件激活
    if ($UseMailConfirm == false AND $makeAdminOnlyActivate == false){
        // 则将激活码设置为 0，注册账号自动被激活
        $actnum = "0";
    }else{
        // 产生一个随机数作为激活码
        if ($TripodSupport == true){
            $actnum = rand(10000000,99999999999999);
        }else{
            // define the characters that may be in the code
            $chars_for_actnum = array ("A", "B", "C", "D", "E", "F", "G", "H", "I",
"J", "K", "L", "M", "N", "O", "P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y",
"Z","a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m", "n", "o",
"p", "q", "r", "s", "t", "u", "v", "w", "x", "y", "z","1","2","3","4","5","6",
```

```
"7","8","9","0");
            // take 20 times (1 to 20) an random char and add it to the $actnum
variable
            for ($i = 1; $i <= 20; $i++) {
                $actnum = $actnum . $chars_for_actnum[mt_rand (0, count ($chars_
for_actnum)-1)];
            }
        }
    }
    //判断用户名是否合法
    $UserNameChars = "^[A-Za-z0-9_-]";
    $UserNameGood = true;
    if(!ereg("$UserNameChars",$username1))
    {
            $UserNameGood = false;  //不合法设定为 false
    }
    //判断密码是否合法
    $PassWordChars = "^[A-Za-z0-9_-]";
    $PassWordGood = true;
    if(!ereg("$PassWordChars",$password1))
    {
            $PassWordGood = false; //不合法，设定为 false
    }
    $error = ""; //初始化错误为空
    //检查用户名、密码以及电子邮件是否为空
    if ($username1 == ""){$error = "$error<li>No username given<BR>\n";}
    if ($password1 == ""){$error = "$error<li>No password given<BR>\n";}
    if ($mailadres1== ""){$error = "$error<li>No mailadres given<BR>\n";}
    // 错误提示
    if ($UserNameGood == false && $UsernameValCharOnly == true){$error = "$error
<li>The username contains invalid chars.<BR>\n";}
    if ($PassWordChars == false && $passwordValCharOnly == true){$error = "$error
<li>The password contains invalid chars.<BR>\n";}
    if (strlen($password1) < $passwordLengthMIN ){$error = "$error<li>The password
contains to little chars.<BR>\n";}
    if (strlen($password1) > $passwordLengthMAX ){$error = "$error<li>The password
contains to much chars.<BR>\n";}
    if (strlen($username1) < $usernameLengthMIN ){$error = "$error<li>The username
contains to little chars.<BR>\n";}
    if (strlen($username1) > $usernameLengthMAX ){$error = "$error<li>The username
contains to much chars.<BR>\n";}
    //检查 Email 是否合法
    if (validadres($mailadres1) == false ){$error = "$error<li>The given e-mail
adres is not valid<BR>\n";}
    //确定两次密码是否一致
    if ($password1 <> $confirmpassword1) {$error = "$error<li>Passwords do not
match<BR>\n";}
    // 判断用户名和 Email 是否已经存在
    $query = "Select * from ".$DBprefix."signup where username='$username1' or
mailadres='$mailadres1'";
    $result = mysql_query($query);
    if ($row = mysql_fetch_array($result)){
```

```
        if  ($row["username"] == $username1){$error = "$error<li>Your username is
already used by another member<br>\n";}
        if  ($row["mailadres"] == $mailadres1){$error = "$error<li>Your e-mail
adres is already registrated in our database<br>\n";}

    }
    // 接下来处理注册数据
    if ($error == ""){  //如果没有错误
        // 获得时间
        $datetime = date("d-m-Y G:i ");
        //数据连接在 config.php 中创建
        $query = "INSERT INTO ".$DBprefix."signup (username, password, mailadres,
actnum, userlevel, signupdate ,lastlogin, lastloginfail, numloginfail) VALUES
('$username1','$password1','$mailadres1','$actnum', '1', '$datetime', '0', '0',
'0')";
        $result = mysql_query($query);
        // 如果需要电子邮件认证
        if ($UseMailConfirm == true){
            $message = $email_message_content;
            $message = str_replace("%p",$actnum, $message);
            $message = str_replace("%a","%", $message);
            //发送电子邮件到客户端
    mail($mailadres1, $email_message_title, $message, $email_message_header);
        }
        //如果不需要电子邮件认证，则回到登录界面
        if ($UseMailConfirm == false AND $makeAdminOnlyActivate == true){
            header("Location: login.php");
        }else{
            //否则进入到输入验证码界面
            header("Location: activate.php");
        }
    }
    else
    {  //输出错误
        echo "You could not be added to the database because of the following
reason(s)<ul>
        $error
        </ul>Please return to <a href=\"signup.php\">signup form</a> and try
again.";
    }

    function validadres($MailAdres){
        //验证电子邮件是否合法，省略代码
    }
    ?>
```

在 add.php 中，首先要产生一个激活码，该激活码会在用户注册成功后发送到用户注册的电子邮箱中。当用户获取到激活码之后，输入激活码，才能使得注册账号发挥作用。否则用户即使注册了，也不能登录。另外，在注册时候，需要判断新注册用户名是否已经存在，如果存在，则提示用户重新输入。

发送电子邮件是通过 mail 函数来实现的，有关发送电子邮件的详细信息，请参照“电子

邮件收发系统”一章。

13.3.2　账号激活模块

注册完账号之后，系统会自动引导到账号激活页面，如图 13-2 所示。

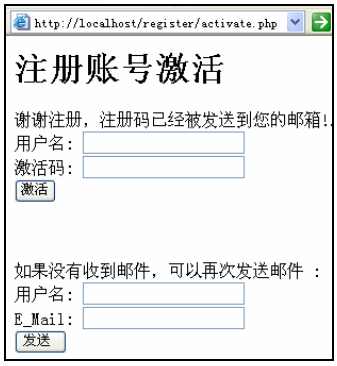


图 13-2　激活账号

如果用户没有收到激活码，可以让系统再次发送。实现账号激活的文件名为 activate.php，如代码 13-5 所示。

【代码 13-5】　activate.php

```
   <?PHP
   //获取用户名、验证码和邮件地址
   $username1 = $HTTP_GET_VARS["username"];
   $actnum1 = $HTTP_GET_VARS["actnum"];
   $mailadres1 = $HTTP_GET_VARS["mailadres"];
   // 检查是否为重发验证码
   $resend = $HTTP_GET_VARS["resend"];
   //包含文件
   include("config.php");
   //如果是再次发送，并且需要邮件发送验证码
   if ($resend == "1" && $UseMailConfirm = true){
       //检查管理员是否已经禁止了该功能
       if ($allowResend == true){
           $query = "Select * from ".$DBprefix."signup where username='$username1'
And mailadres='$mailadres1'";
           $result = mysql_query($query);
           //获取数据
           if ($row = mysql_fetch_array($result)){
               //获取验证码
               $actnum = $row["actnum"];
               $message = $email_message_content;
               $message = str_replace("%p",$actnum, $message);
               $message = str_replace("%a","%", $message);
               // 发送到用户电子邮箱中
   mail($mailadres1, $email_message_title, $message, $email_message_header);
               //再次发送成功
               makeform($activationCodeHasBeenResend);
               //退出
```

```
                die();
            }else{
                // 用户名或者电子邮件错误
                makeform($incorrectUserMailaders);
                die();
            }
        }else{
            // 功能被限制
            makeform($disabledFeatures);
            // 退出
            die();
        }
    }
    // 检查用户名和激活码是否正确
    $query = "Select * from ".$DBprefix."signup where username='$username1' And
actnum='$actnum1'";
    $result = mysql_query($query);
    if ($row = mysql_fetch_array($result)){   //激活后，将激活码清 0
        $query = "UPDATE ".$DBprefix."signup Set actnum = '0' where username=
'$username1'";
        $result = mysql_query($query);
    ?>
    <html>
    <head>
    <title>Activate account</title>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
    </head>
    <body>
    <h1>Sign-Up Complete</h1>
    <p>Thank you for activating your account. You may now <a href="login.php">
login</a>
    </p>
    </body>
    </html>
    <?PHP
    }
    //如果输入错误
    else
    {
        //如果用户名为空
        if ($username1 == ""){makeform();}
        //重新显示激活表单，并显示提示
        else
        {
            makeform($incorrectUserActcode);
        }
    }
    // 显示激活表单的函数
    function makeform($errormessage=""){
    ?>
    <html>
    <head>
```

```
    <title>Activate account</title>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
    </head>
    <body>
    <h1>注册账号激活</h1>
    <form name="form1" method="get" action="activate.php">
    <?PHP
    //检查是否有错误信息
    if ($errormessage != ""){
        echo "<font color=\"#FF0000\"><strong>$errormessage</strong> </font>
<br><br>";}
    ?>
      谢谢注册，注册码已经被发送到您的邮箱!.<br>
      用户名:
      <input name="username" type="text" id="username">
      <br>
      激活码:
      <input name="actnum" type="text" id="actnum">
      <br>
      <input type="submit" value="激活">
    </form>
    <br>
    <br>
    <form name="form2" method="get" action="activate.php">
    如果没有收到邮件，可以再次发送邮件 :<br>
      用户名:
      <input name="username" type="text" id="username">
      <br>
      E_Mail:
      <input name="mailadres" type="text" id="mailadres">
      <input name="resend" type="hidden" id="resend" value="1">
      <br>
      <input type="submit" value="发送 ">
    </form>
    </body>
    </html>
    <?PHP
    }
    ?>
```

在激活页面，用户需要输入从邮箱中输入的激活码，然后激活。激活后，数据库中的激活码被置 0，这样用户就可以成功登录了。如果用户没有收到注册码，则可以让系统再次发送邮件，注意邮箱必须是注册的时候用到的邮箱，而不能随意输入一个其他邮箱，这样的目的是为了保证用户的安全性。

13.3.3　系统登录模块

一旦激活账号之后，就可以利用账号登录了。在浏览器中输入“http://localhost/register/login.php”，即可进入登录界面，如图 13-3 所示。

如果用户选择了“保存登录信息”复选框，则在 7 天之内，链接该地址，系统会自动登录。登录成功后会进入其他管理模块。

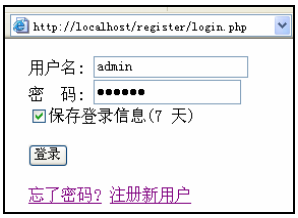


图 13-3　系统登录

【代码 13-6】　login.php

```
    <?PHP
    // 获取用户的登录信息
    $username1 = $HTTP_POST_VARS["username"];
    $password1 = $HTTP_POST_VARS["password"];
    $rememberMe = $HTTP_POST_VARS["rememberMe"];
    // 判断用户是否选择了保存密码
    if ($rememberMe == "rememberMe"){
        $rememberMe = "1";
    }else{
        $rememberMe = "0";
    }
    // 包含设置文件
    include("config.php");
    //检查用户名是否存在
    $query = "Select * from ".$DBprefix."signup where username='$username1'";
    $result = mysql_query($query);
    if ($row = mysql_fetch_array($result)){
        // 检查该用户账号是否已经激活
        if ($row["actnum"] == "0"){
            //检查账号是否被锁定
            if ($row["numloginfail"] <= 5){  //如果登录失败大于 5 次，则自动锁定
                //检查密码是否正确
                if ($row["password"] == $password1){
                    //最后登录时间
                    $datetime = date("d-m-Y G:i ");
                    //修改记录
                    $query = "UPDATE ".$DBprefix."signup Set lastlogin =
'$datetime' where username='$username1'";
                    $result = mysql_query($query);
                    // 清空登录失败记录，设置为 0
                    $query = "UPDATE ".$DBprefix."signup Set numloginfail = '0'
where username='$username1'";
                    $result = mysql_query($query);
                    // tell we want to work with sessions
                    session_start();// 启动会话
                    session_unset();//删除会话
                    session_destroy();
                    session_register("pass");  //创建会话变量，保存密码
                    $HTTP_SESSION_VARS["pass"] = $password1;
                    session_register("id");  //保存用户名
                    $HTTP_SESSION_VARS["id"] = $username1;
```

```
                    // 发送 cookie 到客户端，密码被加密
                    if($rememberMe=="1"){
                    setcookie("rememberCookieUname",$username1,(time()+604800));
                    setcookie("rememberCookiePassword",md5($password1),(time()
+604800));
                    }
                    // go to the secured page.
                    header("Location: members/index.php");
                }
                else{
                    // 密码错误，登录失败
                    // 改变当前时间的格式
                    $datetime = date("d")*10000000000 + date("m")*100000000 +
date("Y")*10000 + date("G")*100 + date("i");
                    //检查上一次登录失败时间是否为 5min 内
                    if ($row["lastloginfail"] >= ($datetime-5)){
                        // 如果是，则将登录失败次数增加 1
                        $query = "UPDATE ".$DBprefix."signup Set numloginfail =
numloginfail + 1 where username='$username1'";
                        $result = mysql_query($query);
                        //修改登录失败时间
                        $query = "UPDATE ".$DBprefix."signup Set lastloginfail =
'$datetime' where username='$username1'";
                        $result = mysql_query($query);
                    }
                    else{
                        // 如果 5min 之内，则只修改登录失败的时间
                        $query = "UPDATE ".$DBprefix."signup Set lastloginfail =
'$datetime' where username='$username1'";
                        $result = mysql_query($query);
                    }
            // 输出登录错误信息
            makeform($incorrectLogin);}
            }
            // 如果登录失败 5 次，表示有人试图猜测密码
            // 因此需要检查时间，如果最后一次登录失败到现在间隔超过半小时，则自动解锁
            // 然后需要重新登录一次
            else {
                $datetime = date("d")*10000000000 + date("m")*100000000 +
date("Y")*10000 + date("G")*100 + date("i");
                if ($row["lastloginfail"] <= ($datetime-30)){
                    // 如果在半个小时之前，则修改 5，表示解锁
                    $query = "UPDATE ".$DBprefix."signup Set numloginfail = '5'
where username='$username1'";
                    $result = mysql_query($query);
                    //显示表单让用户重新登录
                    makeform($underAttackReLogin, "$username1");
                }
                else{
                // 如果在 30min 之内，则仍然被锁定
                    echo $underAttackPleaseWait;
                }
```

```
            }
        }
        //如果账号没有被激活
        else{
            makeform($accountNotActivated);
        }
    }
    // 如果用户名不存在
    else{
        // 检查用户是否输入用户名
        if ($username1 == ""){
            makeform("");  //重新显示登录表单
        }
        else {  //如果输入了，则提示用户名输入错误
            makeform($incorrectLogin);
        }
    }
    //显示登录表单的函数
    function makeform($errormessage="", $username2 = ""){
        echo $errormessage;
    ?>
    <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
    <html>
    <head>
    <title>LOG-IN</title>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
    </head>
    <body>
    <h1 系统登录 </h1>
    <?PHP
    if ($errormessage != ""){echo "<font color=\"#FF0000\"><strong>$errormessage
</strong></font><br>";} ?>
    <form name="form1" method="post" action="login.php">
      <p>用户名:
        <input name="username" type="text" id="username" value=<?PHP
         echo "\"$username2\"";
         if ($username2 != ""){echo "DISABLED";}
    ?>>
        <br>
        密  码:
        <input name="password" type="password" id="password"><br>
         <input type="checkbox" name="rememberMe" value="rememberMe">保存登录信息
(7 天)<br>
        <br>
        <input type="submit" value="登录">
      </p>
    </form>
    <a href="forgot.php">忘了密码?</a>  <a href="signup.php">注册新用户</a>
    </body>
    </html>
    <?php } ?>
```

系统登录模块，首先需要查询登录的用户名是否已经注册，然后检查该用户名是否被激活，如果没有被激活，则不能登录成功。如果用户设置了保存密码，则设置一个 cookie 来保存用户名和密码，为了安全，密码进行 md5 加密。

为了防止用户猜测密码，当用户连续 5 次输入错误密码之后，系统会自动将该账号锁定，用户只能半个小时后，才能继续登录。

13.3.4　取回密码模块

如果用户忘记了密码，可以在登录界面中单击“忘记密码”链接，直接进入取回密码界面，如图 13-4 所示。



图 13-4　找回密码界面

用户输入用户名以及注册时输入的 Email 地址，单击“发送密码”按钮，密码会被发送到客户邮箱。通常找回密码的方法有很多，有些是增加提问，比如在注册的时候让用户输入“密码提示问题”以及“密码提示答案”。由于很多用户在注册的时候，可能随机的写了一个，实际密码丢失的时候不会想起来。而通过将密码发送到自己邮箱的方法是比较可靠的找回密码的方法。

取回密码由文件 forgot.php 文件来实现，如代码 13-7 所示。

【代码 13-7】　forgot.php

```
    <?PHP
    // 获取用户名和信箱
    $username1 = @$HTTP_POST_VARS["username"];
    $mailadres1 = @$HTTP_POST_VARS["mailadres"];
    //包含文件
    include("config.php");
    // 检查管理员是否允许取回密码
    if ($AllowForgotPassword == false){die($disabledFeatures);}
    // 检查用户名和电子邮件是否正确
    $query = "Select * from ".$DBprefix."signup where username='$username1' And
mailadres='$mailadres1'";
    $result = mysql_query($query);
    // 如果正确，则获取密码发送到客户端
    if ($row = mysql_fetch_array($result)){
        $message = $message_forgot_password;
        $message = str_replace("%p",$row["password"], $message);
        $message = str_replace("%a","%", $message);
        mail($mailadres1, $title_forgot_password, $message, $email_message_header);
        //显示密码被发送成功的信息
        echo "The password has been send to your e-mailades. You should recieve
it within a couple of minutes.<br><a href=\"login.php\">back to log-in page</a>";
        }
```

```
    else{
        //如果用户名和电子邮件错误
        if ($username1 == ""){makeform();}
        else
        // 显示错误提示信息
        {makeform($incorrectUserMailaders);}
    }
    // 创建发送密码的表单
    function makeform($errormessage=""){
    ?>
    <html>
    <head>
    <title>Forgot password</title>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
    </head>
    <body>
    <h1>找回密码</h1>
    <form name="form1" method="post" action="forgot.php">
      <a href="login.php">返回登录页面</a><br>
      <?PHP
    // print the error message
    echo "<font color=\"#FF0000\"><strong>$errormessage</strong></font><br>"; ?>
      用户名:
      <input type="text" name="username">
      <br>
      E_Mail:
      <input type="text" name="mailadres">
      <br>
      <input type="submit" value="发送密码">
    </form>
    </body>
    </html>
    <?PHP } ?>
```

取回密码相对比较简单，只需要让用户输入用户名和邮箱，系统便将密码发送到邮箱中。邮箱必须是用户注册时使用的邮箱，否则系统不会发送密码。

13.3.5 修改密码模块

修改密码也是登录系统中的一个重要组成部分。大部分的系统都会提供这种服务，实现方法有两种，一种是先登录，由系统功能菜单中提供的修改密码的链接进入修改密码功能模块。另外一种是直接在登录界面上提供修改密码链接进入修改密码模块。

此处采用前者，即登录后提供修改密码菜单，如图 13-5 所示。

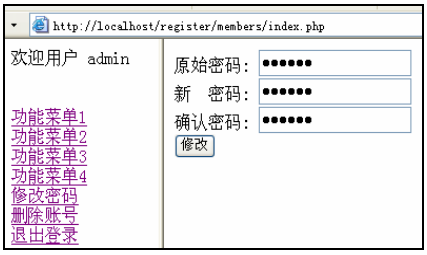


图 13-5 修改密码界面



在让用户修改密码的时候，必须要让用户输入原始密码，这样防止被别有用心的人乘机修改。

在图 13-5 中，单击“修改密码”链接，可以修改密码。修改密码由文件 changepass.php 实现，如代码 13-8 所示。

【代码 13-8】　changepass.php

```
    <?PHP
    $reqlevel = 1;
    include("membersonly.inc.php");
    // 检查是否允许修改密码
    if ($allowChangePassword == false){die($disabledFeatures);}
    //获取原始密码和新密码
    $oldpass1 = $HTTP_POST_VARS["oldpass"];
    $password1 = $HTTP_POST_VARS["password"];
    $password2 = $HTTP_POST_VARS["password2"];
    //检查两个密码是否一致，同时检查原始密码是否为空
    if ($password1 == $password2 && $password1 <> ""){
        //检查原始密码是否正确
        $query = "Select * from ".$DBprefix."signup where username='$HTTP_
SESSION_VARS[id]' And password='$oldpass1'";
        $result = mysql_query($query);
        if ($row = mysql_fetch_array($result)){
            //修改密码
            $query = "UPDATE ".$DBprefix."signup Set password = '$password1' where
username='$HTTP_SESSION_VARS[id]'";
            $result = mysql_query($query);
            //同时修改会话变量的密码
            $HTTP_SESSION_VARS["pass"] = $password1;
            //echo an confirm.
            echo "You password was changed ";
        }else{
            //否则提示原始密码错误
            makeform("Your old password is not correct. pleast try again");
        }
    }else{
        if ($password1 == ""){makeform("");}
        else{makeform("You password do not match try again.");}
    }
    //显示修改密码表单
    function makeform($errormessage){
    ?>
    <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
    <html>
    <head>
    <title>Change password</title>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
    </head>
    <body>
    <?PHP echo "<font color=\"#FF0000\"><strong>$errormessage </strong>
```

```
</font>";?>
    <form name="form1" method="post" action="changepass.php">
      <p>原始密码:
        <input name="oldpass" type="password" id="oldpass">
        <br>
        新  密码:
        <input name="password" type="password" id="password">
         <br>
        确认密码:
        <input name="password2" type="password" id="password2">
        <br>
        <input type="submit" value="修改">
      </p>
    </form>
    </body>
    </html>
    <?php } ?>
```

13.3.6 注销登录模块

用户登录成功后，系统会一直跟踪用户的状态。如果用户在规定时间内没有任何操作，则系统会自动注销。如果用户登录网站后，希望消除自己的登录信息，可以采用网站提供的注销登录链接进行注销，这样可以减少资料被人窃取的可能。特别是使用完网络银行之后，用户应该注销登录，而不应该直接关闭浏览器或者直接进入别的网站。

在图 13-5 中，单击链接“退出登录”即可以注销用户。注销用户登录是由文件 logoff.php 实现的，如代码 13-9 所示。

【代码 13-9】 logoff.php

```
    <?PHP
    //启动会话
    session_start();
    //设置 cookie
    setcookie("rememberCookieUname","session expired",(time()+604800));
    setcookie("rememberCookiePassword","X@X",(time()+604800));
    session_unset(); //删除会话
    //remove the session itself
    session_destroy();
    //回到登录界面
    header("Location: login.php");
    ?>
```

13.4 验证码的实现

验证码是最近几年才普遍应用的。以前，用户登录网站的时候，只需要提供用户名和密码即可。但是随着网络的发展，网络安全受到了人们的重视，因此很多的安全手段也产生了，验证码就是其中之一。

所谓验证码就是用户在登录的时候，要输入一个系统随机产生的字符串序列，和用户名以及密码一同提交。

图 13-6 所示显示的是某个网站的验证码输入界面。

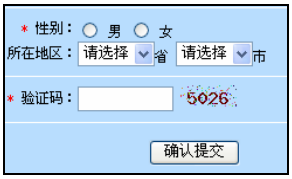


图 13-6　验证码输入界面

验证码的实现原理就是在服务器端产生一个随机字符串，然后保存在服务器中，以图形的方式发送到客户端，客户端在登录的时候输入验证码，如果和服务器保存的验证码一致，则通过验证，如果不一致，则需重新产生一个验证码发送到客户端。每次用户登录的验证码都是随机产生的，使用一次之后失效。

这里介绍一个验证码产生系统，该系统由 3 个文件组成，如下所示：

（1）index.php，将验证码发送到客户端，并显示登录界面；

（2）verificate.php，用于验证用户输入的验证码是否正确；

（3）doimg.php，将验证码以图形的方式发送到浏览器。

13.4.1　将验证码发送的客户端（index.php）

该文件用来显示验证码。将本系统的目录“verifycode”拷贝到服务器根目录，然后在浏览器中输入“http://localhost/verifycode/index.php”，即可登录到该验证码产生系统。界面如图 13-7 所示。



图 13-7　输出验证码

index.php 文件如代码 13-10 所示。

【代码 13-10】　index.php

```
<?php
    //启动会话，将产生的验证码保存到会话变量中
    session_start();
    // 初始化变量
    $IMGVER_TempString="";
    //产生 6 个随机字符
    for ($i = 1; $i <= 6; $i++) {
        $IMGVER_TempString .= GetRandomChar();
    }
    // 将产生的随机字符串保存到会话变量中
```

```
        $_SESSION["IMGVER_RndText"] = $IMGVER_TempString;
        //echo $_SESSION["IMGVER_RndText"];
        function GetRandomChar() {
        // 调用 mt_srand 产生随机数
        mt_srand((double)microtime()*1000000);
        //创建一个 1～3 之间的数字
        $IMGVER_RandVal = mt_rand(1,3);
        //如果产生的随机数组为 1，则产生一个小写的字符
        //如果为 2，则产生一个数字
        //如果为 3，则产生一个大写字符
            switch ($IMGVER_RandVal) {
           case 1:
               //  ASCII 值为 97～122 表示小写字符，即 a～z
              $IMGVER_RandVal = mt_rand(97, 122);
              break;
           case 2:
             //  ASCII 值为 48～57 表示数字，即 0～9
              $IMGVER_RandVal = mt_rand(48, 57);
              break;
           case 3:
               //  ASCII 值为 65～70 表示大写字符，即 A～Z
              $IMGVER_RandVal = mt_rand(65, 90);
              break;
            }
           // 返回创建的随机字符
            return chr($IMGVER_RandVal);
        }
    ?>
    <table width="500" border="0" cellspacing="0" cellpadding="0">
      <tr>
        <td><p><font size="2" face="Verdana, Arial, Helvetica, sans-serif">产生
验证码<br />

          <hr size="1" noshade="noshade" />
          <p><font size="2" face="Verdana, Arial, Helvetica, sans-serif">输入验证
码后，单击 OK 按钮提交</font></p>
          <p align="center"><font size="2" face="Verdana, Arial, Helvetica,
sans-serif">
           <img src="doimg.php?<?php echo SID ?>" /></font></p>
          <form action="verificate.php" method="POST" name="frmImgVerific"
target="_self" id="frmImgVerific">
            <p> <font size="2" face="Verdana, Arial, Helvetica, sans-serif">请输
入验证码：</font>
              <input name="txtCode" type="text" id="txtCode" size="30" />
            </p>
            <p>
              <input type="submit" name="Submit" value="OK" />
            </p>
          </form>
          <p align="center"><font size="2" face="Verdana, Arial, Helvetica,
sans-serif">
             <B>注意：</b>输入的字符是区分大小写的!
```

```
        </p>
      </td>
    </tr>
  </table>
```

index.php 文件主要生成一个随机数，然后保存在 session 变量中。而该随机数是以图形方式显示的，调用 doimg 文件来显示。

13.4.2　verificate.php

verificate.php 文件判断输入的验证码是否正确，如果错误，则重新返回到登录页面，如果正确则提示登录成功，如代码 13-11 所示。

【代码 13-11】　verificate.php

```
<?php
    //启动会话
    session_start();
    // 获取用户验证码以及系统产生的验证码
    // the random generated text
    $IMGVER_EnteredText = $_POST["txtCode"];
    $IMGVER_RandomText = $_SESSION["IMGVER_RndText"];
    //如果相等
    if ($IMGVER_EnteredText == $IMGVER_RandomText) {
        echo "登录成功!";
                /* ENTER YOUR MESSAGE ABOVE OR MAKE THIS FILE A FUNCTION
                ** AND USE RETURN INSTEAD! */
    } else {
        echo "登录失败，请重新登录";
        echo "<a href=index.php>回到登录页面</a>";
                /* ENTER YOUR ERROR MESSAGE ABOVE OR MAKE THIS FILE A
FUNCTION
                ** AND USE RETURN INSTEAD! */
    }
    //删除会话
    session_unset();
    session_destroy();
?>
```

13.4.3　doimg.php

doimg.php 文件用于将产生的验证码以图形方式输出，如代码 13-12 所示。

【代码 13-12】　doimg.php

```
<?php
// 创建图像区域
$IMGVER_IMAGE = imagecreate(150,40);
//创建颜色
$IMGVER_COLOR_BLACK = imagecolorallocate ($IMGVER_IMAGE, 0, 0, 0);
$IMGVER_COLOR_WHITE = imagecolorallocate ($IMGVER_IMAGE, 255, 255, 255);
//黑色填充
imagefill($IMGVER_IMAGE, 0, 0, $IMGVER_COLOR_BLACK);
```

```
    //启动会话
    session_start();
    //获取产生的随机字符串
    $IMGVER_RandomText =$_SESSION["IMGVER_RndText"];
    //依次显示 6 个字符
    imagechar($IMGVER_IMAGE,4,20,13,$IMGVER_RandomText[0] ,$IMGVER_COLOR_WHITE);
    imagechar($IMGVER_IMAGE, 5, 40, 13, $IMGVER_RandomText[1] ,$IMGVER_COLOR_WHITE);
    imagechar($IMGVER_IMAGE, 3, 60, 13, $IMGVER_RandomText[2] ,$IMGVER_COLOR_WHITE);
    imagechar($IMGVER_IMAGE, 4, 80, 13, $IMGVER_RandomText[3] ,$IMGVER_COLOR_WHITE);
    imagechar($IMGVER_IMAGE, 5, 100, 13, $IMGVER_RandomText[4] ,$IMGVER_COLOR_WHITE);
    imagechar($IMGVER_IMAGE, 3, 120, 13, $IMGVER_RandomText[5] ,$IMGVER_COLOR_WHITE);
    //显示图像
    header("Content-type: image/jpeg");
    imagejpeg($IMGVER_IMAGE);

    ?>
```

至此，有关注册、登录以及验证码的关键技术和代码就介绍完了，读者可以根据光盘代码进行适当的扩展，以编写出适合自己需求的系统。

第14章　投 票 系 统

本章将介绍如何在 PHP 网络中实现投票系统。投票系统一个完整的网站系统中不可缺少的一部分。

作为一个投票系统，必须具备以下几个最基本的功能：

（1）提供多种选项；

（2）让用户进行选择（可以根据需要进行单选或者多选，通常为单选）；

（3）显示投票结果。

网站管理员必须能够具备以下的管理的权限：

（1）创建投票主题；

（2）创建投票选项；

（3）删除、编辑投票选项；

（4）让用户选择不同的投票主题；

（5）限制用户投票的次数。

以上是一个投票系统必须具备的功能，当然可以进行更多的扩展。本章将介绍两个投票系统，一个是基于数据库的，另外一个是基于文本文件的。

14.1　基于数据库的投票系统

首先介绍投票系统的文件组成结构，然后介绍数据库创建步骤，最后介绍系统的实现。

14.1.1　系统整体设计

投票系统是一个小型的系统，涉及的文件数量并不多，具体文件说明如下。

（1）show.php，用于显示被投票的主题以及选项。管理员可以预先设定多个投票主题，只有被激活的主题才会显示出来。

（2）pollresult.php，用于图形化地显示投票结果，包括总投票数、每个选项被投票的百分比。

（3）config.php，用于数据库连接，该文件同时被客户端和管理员使用。

（4）functions.php，该文件定义了一些常用的函数，用于显示投票结果、投票问题等，该文件同时被客户端和管理员使用。

（5）createdb.php，用于创建数据库，为了方便读者使用，这里采用了程序自动创建数据库表的技术。

（6）style.css，用于美化显示的 CSS 文件。

管理员进行管理的，只包含一个文件：admin.php，用于添加、编辑、删除和设定当前投票主题等。

14.1.2 数据库设计

投票系统的后台数据库采用 MySQL 5.0，其中要创建的数据库名称为“poll”。该数据库具有两个数据库表，一个用于保存投票主题信息，表名为 addpoll；另外一个用于显示投票结果，表名为 pollresult。

addpoll 表的设计 SQL 语句如下：

```
CREATE TABLE addpoll
(pid tinyint(4) NOT NULL auto_increment,  //投票主题 ID
question varchar(100),    //投票主题内容
answer1 varchar(40),      //投票选项，每个投票主题最多 6 个投票选项
answer2 varchar(40),
answer3 varchar(40),
answer4 varchar(40),
answer5 varchar(40),
answer6 varchar(40),
noofanswers tinyint(1),
display tinyint(1) DEFAULT '0',     //表示是否为当前激活主题，默认为 0，激活为 1
PRIMARY KEY (pid)      //设置 pid 为主键
);
```

pollresult 表设计的 SQL 语句如下：

```
CREATE TABLE pollresult (
pid int(4),    //投票主题编号
answer1 int(4),    //以下为每个投票选项被投票的次数
answer2 int(4),
answer3 int(4),
answer4 int(4),
answer5 int(4),
answer6 int(4),
noofanswers tinyint(4)
);
```

介绍完数据库设计之后，接下来需要在 MySQL 中创建数据库以及数据库表。关于数据库的创建将在下一小节“系统实现”中介绍。

14.1.3 系统实现

在分析了数据库之后，接下来实现整个系统。本系统的数据库表是通过浏览器执行数据库创建文件而实现的。

1．创建数据库以及数据库表

首先使用记事本或者其他文本编辑器，如 Dreamweaver 等，打开 config.php 文件，如代码 14-1 所示。

【代码 14-1】 config.php

```
<?php
/////// 修改数据库连接设置/////////////////////////////////////////
        $host="localhost";
        $user="root";                //用户名
        $password="123456";         //密码
        $database="poll";            //创建的数据库名称
```

```
    ////////////////////////////////////////////////////////////////////////
    //连接数据库服务器
    $dblink= mysql_connect("$host","$user","$password");
    //列出数据库
    $listdbs = mysql_list_dbs($dblink);
    $dbarray=array();
    while ($databaseObject = mysql_fetch_object($listdbs)) {
        array_push($dbarray,$databaseObject->Database);
    }
    if (!in_array($database,$dbarray)){
        //如果数据库中没有 POLL 数据库，则创建
        $sql="create database $database";
        mysql_query($sql,$dblink);
    }
    //创建一个连接指定数据库的连接
    function dbconnect($host,$user,$password,$database){
        //创建一个持久数据库连接
        mysql_pconnect("$host","$user","$password") or die ("登录失败!");
        //选择数据库
        mysql_select_db($database) or die ("failed connection to database");
        echo mysql_error();
    }
    //执行函数
    dbconnect($host,$user,$password,$database);
    ?>
```

在代码 14-1 中，主要设置数据库连接参数，包括用户名、密码以及要创建的数据库名称，默认为“poll”，建议不要修改。

进行基本设置之后，接下来调用 createdb.php 文件创建整个数据库，如代码 14-2 所示。

【代码 14-2】　createdb.php

```
    <?php
    //包含数据设置文件
    include_once("config.php");
    //创建 SQL 语句
    $queryArr=array("
    CREATE TABLE addpoll (pid tinyint(4) NOT NULL auto_increment,question
varchar(100),answer1 varchar(40),answer2 varchar(40),answer3 varchar(40),answer4
varchar(40),answer5 varchar(40),answer6 varchar(40),noofanswers tinyint(1),display
tinyint(1) DEFAULT '0',PRIMARY KEY (pid));",
    "INSERT INTO addpoll VALUES ( '1', 'Howz this coding', 'Very regular', 'Good
but can be better', 'Not that Bad enough', NULL, NULL, NULL,'3', '1');",
    "CREATE TABLE pollresult (pid int(4),answer1 int(4),answer2 int(4),answer3
int(4),answer4 int(4),answer5 int(4),answer6 int(4),noofanswers tinyint(4));",
    "INSERT INTO pollresult VALUES ( '1', NULL, '1', NULL, NULL, NULL, NULL,
'3');");

    //循环读取每一句 SQL 语句
    foreach($queryArr as $values){
        mysql_query($values);
    }
```

```
    echo "<center><b>数据库和表创建完毕!</b></center><br>";
    ?>
    <a href=admin.php>管理员<a>
    <a href=show.php>开始投票<a>
```

执行 createdb.php 文件（本系统的目录名称为 poll_1，读者可以将其拷贝到系统的根目录下，在前面介绍安装时，根目录为“d:\masterphp”，因此此时目录为“d:\masterphp\poll_1”），在浏览器中输入“http://localhost/poll_1/createdb.php”，执行结果如图 14-1 所示。

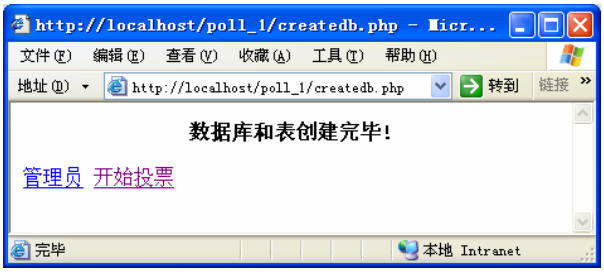


图 14-1　创建数据库

读者此时可以验证数据库是否安装成功，启动 MySQL 客户端程序，如图 14-2 所示。

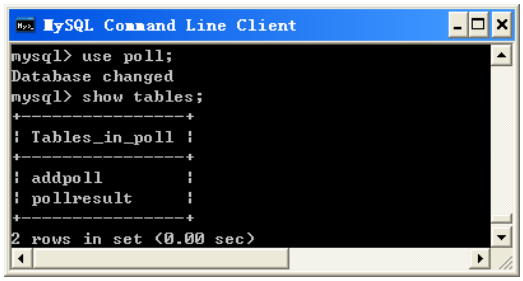


图 14-2　验证数据库是否创建成功

从图 14-2 中可以看出，数据库已经创建成功了。

2．实现管理员功能

管理员功能由文件 admin.php 实现，可以在浏览器中输入“http://localhost/poll_1/admin.php”进入到管理员管理模块，如图 14-3 所示。

单击“添加投票主题”链接时，显示添加界面，如图 14-4 所示。

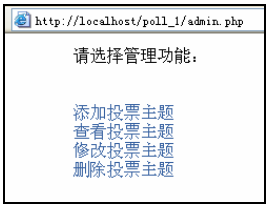


图 14-3　管理功能菜单

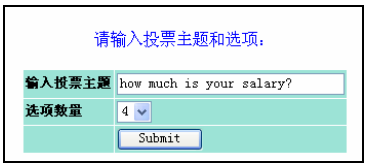


图 14-4　添加投票主题

其中，选项数量可以选择 1～6 个，投票主题是各种需要调查的问题，输入完毕后，单击“submit”按钮，进入选项设计阶段，如图 14-5 所示。

输入选项后，单击“submit”按钮，可以提交到数据库中。

如果在管理主界面中单击“查看投票主题”链接，可以查看所有的投票主题的详细信息，

如主题名称、选项设置，如图 14-6 所示。

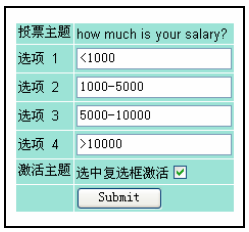


图 14-5　输入投票选项

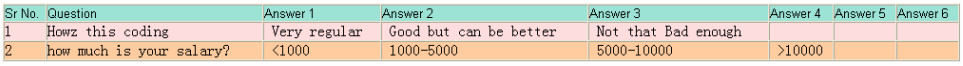


图 14-6　查看投票主题

单击“修改投票主题”链接，可以修改投票主题，其界面同图 14-6 类似，只是每个投票主题后面添加了“modify”的链接，如图 14-7 所示。

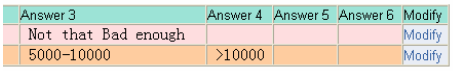


图 14-7　编辑投票主题

单击“modify”链接后，编辑界面同图 14-5 所示类似。

如果要删除投票主题，可以在管理主界面上，单击“删除投票主题”链接，如图 14-8 所示。

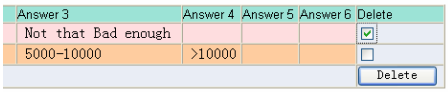


图 14-8　删除投票主题

在图 14-8 中选中被删除的主题，单击“Delete”按钮即可。

管理员功能由文件 admin.php 实现，如代码 14-3 所示。

【代码 14-3】　admin.php

```
    <?php
    include_once("functions.php");  //包含 functions.php 文件
    site_header("Admin for polling");  //显示页面头
    //显示新建主题名称以及选项数量的表单，所有表单的处理文件仍然为 admin.php
    function showForm($error){
    echo "<font class=blue><center>请输入投票主题和选项：";
    echo $error;
    echo "<br>";
    //输出管理员表单，其中提交处理文件为 admin.php
    echo "<form action=admin.php method=post>
          <table action=admin.php method=post align=center cellspacing=1 cellpadding
=2>
          <tr>
            <td class=infoboxtitle>输入投票主题</td>
            <td class=infoboxtitle><input size=30 type=text name='question'></td>
```

```
        </tr>
        <tr>
          <td class=infoboxtitle>选项数量</td>
          <td class=infoboxtitle><select name=noa>";
          for ($i=1; $i<7; $i++){
              if ($noa==$i){
              echo "<option value=$i selected>$i</option>";
              } else {
              echo "<option value=$i>$i</option>";
              }
          }
   echo "</select></td></tr>
       <tr>
           <td class=infoboxtitle> </td>
           <td class=infoboxtitle><input type=submit value=' Submit '></td>
       </tr>
        </table>
        <input type=hidden name=state value=submit>
        </form>";
   }
   //提交处理后的表单显示
   function submit($question,$answer,$noa,$error,$url,$formaction,$recid){
       if ($question=='' || empty($question)){  //如果调查问题为空
       $error="<font color=red><center>Please fill in the question.";
       showForm($error);  //调用 showform 函数显示重新输入调查问题表单
       } else {
           if ($error){
               echo "<font class=red><center>$error</center></font>";
           }
       //如果没有错误，则显示输入调查选项的表单
        echo "<form action=admin.php method=post>";
        echo"<table align=center cellspacing=1 cellpadding=2>
         <tr>
           <td class=infoboxtitle>投票主题</td>
           <td class=infoboxtitle>";
           if ($formaction){
               echo "<input type=text name=question value='$question'>";
           } else {
           echo "<input type=hidden name=question value='$question'>$question";
           }
           echo "</td></tr>";

         for ($i=0; $i<$noa; $i++){
             $disp=$i+1;
             echo "<tr>
                   <td class=infoboxtitle>选项 $disp</td>
                   <td class=infoboxtitle><input type=text name=answer[] value
='$answer[$i]'";
                   echo "'></td>
                   </tr>";
         }
        echo "<tr>
```

```
                <td class=infoboxtitle>激活主题</td>
                <td class=infoboxtitle>选中复选框激活<input type=checkbox name=
activate></td>
              </tr>";
         echo "<tr>
                <td class=infoboxtitle> </td>
                <td class=infoboxtitle><input type=submit value=' Submit '></td>
              </tr>";
          echo "</table>
          <input type=hidden name=state value=";
          if ($formaction){
              echo newstate;
          } else {
              echo enter;
          }
          echo ">
          <input type=hidden name=noa value=$noa>
          <input type=hidden name=recid value=$recid>
          </form>";
        }
    }
    //检查选项，检查选项值是否为空
    function checkAnswer($question,$answer,$noa,$activate,$url){
        foreach ($answer as $key=>$value){
            if ($value=='' || empty ($value)){
                $error='yes';
            }
        }
            if ($error){  //如果有空的选项，则提示输入
            $error="<font class=red><center>Please fill in all
answers.</center></font>";
            submit($question,$answer,$noa,$error,$url,$formaction,$recid);
            } else {  //否则将投票选题信息插入到数据库中
            $sql="insert into addpoll (question,";
                if ($activate){ //是否设置为当前投票主题
                //如果设置为当前主题，则要将其他主题设置为非当前，即将 display 字段设置为 0
                $makesql="update addpoll set display=0";
                mysql_query($makesql);
                    $sql.="display,";
                }
                foreach ($answer as $key => $values){
                    $keys=$key+1;
                    $sql.="answer$keys,";
                }
            $sql.="noofanswers) values ('$question',";
                if ($activate){
                    $sql.="1,";
                }
                foreach ($answer as $key => $values){
                $sql.="'$values',";
                }
            $sql.=count($answer).")";
```

```
            mysql_query($sql);  //执行 sql，插入新的调查问题
            echo "<font class=blue><center>Poll entered successfully</center>
</font>";
            }
    }
    //显示所有主题
    function displayall() {
        showPollList($condition);
    }
    //显示所有主题的删除链接
    function displayallfordelete() {
        showPollList('delete');
    }
    //显示所有主题的编辑链接
    function displayallformodify() {
        showPollList('modify');
    }
    //显示被编辑的投票主题信息
    function displaymodifyrec($recid){
        $sql="select * from addpoll where pid=$recid";
        $result=@mysql_query($sql);
        $question=@mysql_result($result,0,'question');
        $noa=mysql_result($result,0,'noofanswers');
        $answer=array();
        for ($i=1;$i<=$noa;$i++){
            $field="answer$i";
            $getanswer=mysql_result($result,0,$field);
            array_push($answer,$getanswer);
        }
        $formaction='yes';
        submit($question,$answer,$noa,$error,$geturl,$formaction,$recid);
    }
    //编辑主题函数
    function updateindb($recid,$question,$answer,$url,$noa,$activate){
    $sql="update addpoll set question='$question',";
    for($i=1;$i<=count($answer);$i++){
        $newi=$i-1;
        $sql.="answer$i='$answer[$newi]',";
        }
    if ($activate){
        //如果激活一个主题，则其他主题都不能被激活
        $makesql="update addpoll set display=0";
        mysql_query($makesql);
    $sql.="display=1";
    } else {
    $sql.="display=0";
    }
    $sql.=" where pid=$recid";
    mysql_query($sql);
    echo "<font class=blue><b><center>Poll updated successfully</center>
</b></font>";
    }
```

```
    //删除主题函数
    function deletenow($deletearr){
    $sql="delete from addpoll where ";
        foreach($deletearr as $value){
        $sql.="pid=$value or ";
        }
        $sql=substr($sql,0,strlen($sql)-3);
        mysql_query($sql);
    echo "<font class=blue><b><center>Poll deleted successfully</center> </b>
</font>";
    }
    switch($state){ //根据用户 url 地址后面的 state 属性，来调用不同的函数
    case show:showForm($error);
        break;
    case submit:submit($question,$answer,$noa,$error,$url,$formaction,$recid);
        break;
    case enter:checkAnswer($question,$answer,$noa,$activate,$url);
        break;
    case viewpoll:displayall();
        break;
    case deletepoll:displayallfordelete();
        break;
    case modifypoll:displayallformodify();
        break;
    case modifyrecords:displaymodifyrec($recid);
        break;
    case newstate:updateindb($recid,$question,$answer,$url,$noa,$activate);
        break;
    case deletepollrecords:deletenow($deletearr);
        break;
    }
    echo "</body></html>";
    ?>
```

提示　　其中部分函数在 functions.php 文件中声明，在后面会介绍。

3．实现客户端投票功能

客户端投票功能比较简单，即直接从数据库中读取被激活的投票主题，然后显示出所有选项。客户端功能由文件 show.php 实现。在地址栏中输入“http://localhost/poll_1/show.php”即可进行投票界面，如图 14-9 所示。

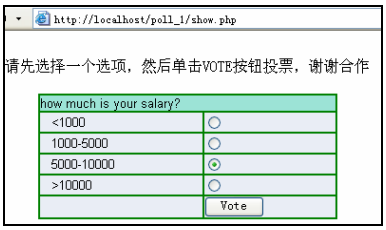


图 14-9　客户端投票界面

单击“Vote”按钮后，数据会被提交到 pollresult.php 文件并显示出投票结果。

【代码 14-4】 show.php

```
    <?php
    include_once("config.php");
    include_once("functions.php");
    echo "<head>
    <title>$title</title>
    <link rel = stylesheet type=text/css href=style.css>
    </head>
    <body bgcolor=#ffffff topmargin=0 leftmargin=0 rightmargin=0>";
    echo "<br><center>请先选择一个选项，然后单击 VOTE 按钮投票，谢谢合作<br>";
    $sql="select * from addpoll where display=1";
    $result=mysql_query($sql);
    $pollRecordArray=mysql_fetch_array($result);
    //表单的处理文件为 pollresult.php
    echo "<form action=pollresult.php method=post>";
    //显示主题名称
    echo "  <table border=1 bordercolor=green align=center cellspacing=0 cellpadding
=0 width=25%>
            <tr>
                <td class=infoboxtitle colspan=2><font class=small>".$pollRecord
Array['question']."</font></td>
            </tr>";
    //循环显示选项
    for($i=1; $i<=$pollRecordArray['noofanswers']; $i++){
        $answer=answer.$i;
    echo "  <tr>
                <td class=infoboxcontent>   <font class=small>".
$pollRecordArray[$answer]."</font></td>
                <td class=infoboxcontent><input type=radio name=answer value=$i>
</td>
            </tr>";
    }
    echo "  <tr>
                <td class=infoboxcontent> </td>
                <td class=infoboxcontent><input type=submit value=' Vote '></td>
            </tr>";
    echo "  </table>";
    //以下均为隐含字段
    echo "<input type=hidden name=id value=".$pollRecordArray[pid].">";
    echo "<input type=hidden name=noa value=".$pollRecordArray[noofanswers].">";
    echo "<input type=hidden name=question value='".$pollRecordArray[question]."'>";
    echo "<input type=hidden name=answer1 value='".$pollRecordArray[answer1]."'>";
    echo "<input type=hidden name=answer2 value='".$pollRecordArray[answer2]."'>";
    echo "<input type=hidden name=answer3 value='".$pollRecordArray[answer3]."'>";
    echo "<input type=hidden name=answer4 value='".$pollRecordArray[answer4]."'>";
    echo "<input type=hidden name=answer5 value='".$pollRecordArray[answer5]."'>";
    echo "<input type=hidden name=answer6 value='".$pollRecordArray[answer6]."'>";
    echo "</form>";
    echo "</body></html>";
    ?>
```

投票页面主要显示被激活的投票主题的标题以及所有选项，供用户选择，选择后的结果将由 pollresult.php 进行处理。

4．显示投票结果

当用户做出选择后，数据被提交到 pollresult.php 文件，该文件将数据保存进数据库，并显示投票结果，如图 14-10 所示。

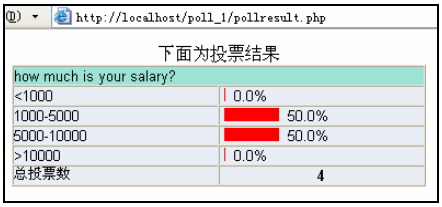


图 14-10　显示投票结果

【代码 14-5】　pollresult.php

```
    <?php
    include_once("config.php");
    include_once("functions.php");
    echo "<head>
    <title>$title</title>
    <link rel = stylesheet type=text/css href=style.css>
    </head>
    <body bgcolor=#ffffff topmargin=0 leftmargin=0 rightmargin=0>";
    echo "<br><center>下面为投票结果<br>";
    if (!$answer){  //如果没有选中一个选项，则回到投票页面
        include_once("show.php");
        exit;
    }
    $answervar="answer".$answer;  //获得选项编号
    //检查投票选项是否存在
    $pollrow=mysql_num_rows(mysql_query("select * from pollresult where pid=$id"));
    if ($pollrow >0){
            //获取该选项被投票的数量
            $getanswer="select $answervar from pollresult where pid=$id";
            $getresult=mysql_query($getanswer);
            $rows=mysql_num_rows($getresult);
            $pollanswer=($rows >0)?mysql_result($getresult,0,$answervar):0;
            //将该数量增加 1
            $pollanswer+=1;
            //更新投票数据到数据库
            $updatesql="update pollresult set $answervar=$pollanswer where pid=$id";
            mysql_query($updatesql);
            //显示结果
           showResult($id,$question,$answer1,$answer2,$answer3,$answer4, $answer5,
$answer6,$answer);
    } else {  //插入新记录，即第一次被投票
            $insertnew="insert into pollresult (pid,$answervar,noofanswers)
values($id,1,$noa)";
            mysql_query($insertnew);
            //显示结果
```

```
            showResult($id,$question,$answer1,$answer2,$answer3,$answer4,$answer5,
$answer6,$answer);
    }
    echo "</body></html>";
    ?>
```

pollresult.php 文件首先判断用户已经做出了选择，如果没有提交选项，则回到投票页面。如果已经做出了选择，则将被选择的投票选项次数增加 1，然后修改数据库记录。如果该选项是第一次被投票，则需要插入一条新记录，然后设置其投票次数为 1。最后调用 showResult 函数显示投票结果。

5．系统中的公共函数

为了减少代码的重复量以及方便代码的阅读和理解，前面的代码中使用了很多公共函数，集中存放在 functions.php 文件中，如代码 14-6 所示。

【代码 14-6】 functions.php

```
    <?php
    include_once("config.php");
    //显示管理功能菜单
    function site_header($title){
        echo "<html>
        <head>
        <title>$title</title>
        <link rel = stylesheet type=text/css href=style.css>
        </head>
        <body bgcolor=#ffffff topmargin=0 leftmargin=0 rightmargin=0>
        <br>
        请选择管理功能：<br><br><br>
        <a href=admin.php?state=show>添加投票主题</a><br>
        <a href=admin.php?state=viewpoll>查看投票主题</a><br>
        <a href=admin.php?state=modifypoll>修改投票主题</a><br>
        <a href=admin.php?state=deletepoll>删除投票主题</a><br>
        ";
    }
    //显示所有投票主题
    function showPollList($condition){
    $sql="select * from addpoll";
    $result=mysql_query($sql);
    $rows=@mysql_num_rows($result);
    if ($rows>0){
    echo "<form action=admin.php method=post>";
    echo "<table width=80% cellpadding=0 cellspacing=0 align=center border=1>";
    echo "<tr>
            <td class=infoboxtitle>Sr No.</td>
            <td class=infoboxtitle>Question</td>
            <td class=infoboxtitle>Answer 1</td>
            <td class=infoboxtitle>Answer 2</td>
            <td class=infoboxtitle>Answer 3</td>
            <td class=infoboxtitle>Answer 4</td>
            <td class=infoboxtitle>Answer 5</td>
            <td class=infoboxtitle>Answer 6</td>";
```

```
            if ($condition==delete){
            echo "<td class=infoboxtitle>Delete</td>";
            }
            if ($condition==modify){
            echo "<td class=infoboxtitle>Modify</td>";
            }
    echo "</tr>";
    $cnt=1;
            while ($recordObj=mysql_fetch_object($result)){
                if ($recordObj->display==1){
                    $bg="bgcolor=#ffcc9f";
                } else {
                    $bg="bgcolor=#ffdddf";
                }
                echo "<tr>";
                echo "<td class=content $bg>".$cnt."</td>";
                echo "<td class=content $bg>".$recordObj->question."</td>";
                echo "<td class=content $bg> ".$recordObj->answer1."</td>";
                echo "<td class=content $bg> ".$recordObj->answer2."</td>";
                echo "<td class=content $bg> ".$recordObj->answer3."</td>";
                echo "<td class=content $bg> ".$recordObj->answer4."</td>";
                echo "<td class=content $bg> ".$recordObj->answer5."</td>";
                echo "<td class=content $bg> ".$recordObj->answer6."</td>";
                    if ($condition==delete){
                        echo "<td class=infoboxcontent><input type=checkbox name
=deletearr[] value=".$recordObj->pid."></td>";
                    }
                    if ($condition==modify){
                        echo "<td class=infoboxcontent><a href=admin.php?state=
modifyrecords&recid=".$recordObj->pid.">Modify</a></td>";
                    }
                echo "</tr>";
                $cnt++;
            }
    if ($condition==delete){
        echo "<tr>
                <td class=infoboxcontent colspan=8> </td>
                <td class=infoboxcontent><input type=submit value=' Delete '></td>
              </tr>";
    }
    echo "</table>";
    echo "<input type=hidden name=state value=deletepollrecords>";
    echo "</form>";
    } else {
    echo "<font class=red><center><b>No poll records exist..<br>Add new poll..
</b></center></font>";
    }
    }
    //显示投票结果
    function showResult($id,$question,$answer1,$answer2,$answer3,$answer4,
$answer5,$answer6,$answer)
    {
```

```
        $answerarr=array();
        array_push($answerarr,$answer1);
        array_push($answerarr,$answer2);
        array_push($answerarr,$answer3);
        array_push($answerarr,$answer4);
        array_push($answerarr,$answer5);
        array_push($answerarr,$answer6);
    echo "  <table width=30% cellpadding=0 cellspacing=0 align=center border=1>
            <tr>
                <td class=infoboxtitle colspan=2><font class=small>$question
</font></td>
            </tr>";
            //获取记录
            $records=showanswers($id);
            $noa=$records[noofanswers];
            for($i=1; $i<=$noa; $i++){
            $answervar="answer".$i;
            $total+=$records[$answervar];
            }
            for($i=1; $i<=$noa; $i++){
            $answervar="answer".$i;
            $arraycnt=$i-1;
            //计算每个投票选项的比例
            $noofvotes=$records[$answervar];
            $width=($noofvotes/$total)*100;
            $height=12;
            ////////  根据用户浏览器来设置投票的结果显示 ////////
            $browserObj=get_browser();
            $user_browser=$browserObj->browser;
            ////////////////////////////////////////////////////////////////////
            if ($width==0){
                if ($user_browser==IE) $height=12;  //设置图形显示高度
                if ($user_browser==Netscape) $height=3;
            }
    //图形化显示投票比例以及文字比例
    echo "  <tr>
            <td class=infoboxcontent width=50%><font class=small>".$answerarr
[$arraycnt]."</font></td>
            <td class=infoboxcontent> <img src=red.gif width=$width height
=$height>  <font class=small>".sprintf("%01.1f",$width)."%</font></td>
            </tr>";
            }
    echo "  <tr>
                <td class=infoboxcontent><font class=small>总投票数 </font></td>
                <td class=infoboxcontent><b><center><font class=small>$total
</font></center></b></td>
            </tr>";

    echo "  </table>";
    }
    //显示选项
```

```
function showanswers($pid){
    $sql="select * from pollresult where pid=$pid";
    $result=@mysql_query($sql);
    return $recordArray=@mysql_fetch_array($result);
}
```

至此，一个基本完善的投票系统已经介绍完毕。细心的读者肯定会注意到，这个投票系统并没有限制用户投票的次数，仔细思考一下该如何实现吧。

14.2　基于文件的投票系统

本节继续介绍投票系统，该投票系统同前面的功能基本类似，但是没有采用数据库，因而使用起来更加方便一些。另外本系统增加了一些限制功能，比如为了体现公平性，限制每台计算机只能投票一次。

14.2.1　系统整体设计

根据对投票系统的功能分析，本投票系统包含了以下的文件。

（1）demo.php，用于让用户选择投票以及进行投票，主要是创建类 Tsvote 实例。

（2）class_vote.php，由于本系统采用了面向对象的编程技术，投票是通过一个类来进行的，该文件主要是进行投票类 Tsvote 的设计。

（3）class_support.php，该文件声明投票类 Tsvote 的基类 SupportFunctions，也就是说 Tsvote 是从 SupportFunctions 类继承而来的。该类实现了一些基本的文件读写的方法。

（4）result_template.php，该文件为一个模板文件，用于显示投票结果，被包含在 class_vote.php 文件中。

（5）choice_template.php，该文件也是一个模板文件，用于显示投票选项，被包含在 class_vote.php 文件中。

（6）vote.css，该文件为整个投票系统的 CSS 样式设计文件。

（7）admin.php，该文件为管理员文件，用于投票主题的管理。

（8）question.dat，该文件保存投票主题以及选项的信息，为文本文件，用普通记事本即可打开。

（9）answer.dat，该文件保存投票结果。

14.2.2　数据的存储形式

本系统没有采用数据库，而是采用了文件的形式。当管理员在管理界面添加一个主题的时候，就自动将投票主题以及选项添加到文件 question.dat 中，示例如下：

```
0|how much is your salary?|1000|2000|3000|4000|
1|你喜欢什么汽车？|君威|雅阁|帕萨特|奥迪|宝马|奔驰|
```

其格式如下：

```
投票主题编号|主题内容|投票选项 1|投票选项 2|……|投票选项 n
```

其中每个数据单元之间用“|”分隔，对于上面的格式，最前面的数字为投票主题的编号、第 2 个单元为投票主题，而后面的单元全部是投票选项。

投票结果则是保存到 answer.dat 文件中，该文件也是一个文本文件，示例如下：

```
0|1|0|1|0|0|
```

```
1|0|0|0|0|0|0|0|
```

该文件的保存格式和 question.dat 类似，每个数据单元用“|”分隔，然后每行的第一个数字表示投票主题的编号，后面的数据则对应每次投票的投票数。如“0|1|0|1|0|0|”中的第 1 个数字 0 表示是第 1 个投票主题（索引从 0 开始）；紧接着是数字 1，表示总投票数是 1；后面 4 个数字分别表示对应选项的投票数量。

其格式如下：

```
投票主题编号|总投票数|选项 1 投票数|选项 2 投票数|……|选项 n 投票数
```

14.2.3 系统实现

接下来介绍投票系统的实现。

1．投票系统的基类 SupportFunctions

SupportFunctions 类保存在 class_support.php 文件中，用于实现一些基本的公共函数。class_support.php 文件如代码 14-7 所示。

【代码 14-7】 class_support.php

```
class SupportFunctions {
    //_writeLine 函数打开一个文件并写入一行数据
    function _writeLine($file, $line, $mode) {
        $fp = fopen($file, $mode) or die("Error while _writeLine to $file");
        if (flock($fp, LOCK_EX + LOCK_NB)) {
            fputs($fp, $line);
            # Take notice of successful execution.
            $logged = 1;
        } else echo("flock failure on $file");
        if (flock($fp, LOCK_UN))
            fclose($fp);
        else echo("flock failure on $file");
        return $logged;
    }
    //_writeArray 函数打开一个文件并写入一个数组
    function _writeArray($file, $array, $mode) {
        # Write an array to a file. The mode is
        # specified like in the fopen function.
        $fp = fopen($file, $mode) or die("Error while _writeArray to $file");
        if (flock($fp, LOCK_EX + LOCK_NB)) {
            for ($i = 0; $i < sizeof($array); $i++) {
                fputs($fp, $array[$i]);
                # Take notice of successful execution.
                $logged = 1;
            }
        } else echo("flock failure on $file");
        if (flock($fp, LOCK_UN))
            fclose($fp);
        else echo("flock failure on $file");
        return $logged;
    }

}
```

2．投票系统的实现类 TSvote

该类从 SupportFunctions 类继承而来，保存在 class_vote.php 文件中，如代码 14-8 所示。

【代码 14-8】　class_vote.php

```
<?php
class TSvote extends SupportFunctions {
        //投票主页名称
    var $pageName = "demo.php";
    var $voteID;
    // 保存数据的文件
    var $question_data = "question.dat";
    var $answer_data = "answer.dat";
    //用于显示
    var $choice_template = "choice_template.php";
    var $result_template = "result_template.php";
    // 用于显示投票结果，不同的选项使用不同的图片
    var $bar_images = array ( "images/aqua.gif",
        "images/yellow.gif",
        "images/red.gif",
        "images/green.gif",
        "images/pink.gif",
        "images/orange.gif",
        "images/darkgreen.gif",
        "images/gold.gif",
        "images/grey.gif",
        "images/blue.gif" );
    //该函数返回关于投票的所有信息，返回一个二维数组
    //该数组包含投票主题、投票选项、投票百分比结果、实际投票数量以及总投票数
    function _getVoteInfo() {
        $question_file = file($this->question_data);
        $answer_file = file($this->answer_data);
        //找到被选择的投票主题
        for ($i = sizeof($question_file) - 1; $i >=0; $i--) {
            $question_line = explode("|", $question_file[$i]);
            if (!strcmp($question_line[0], $this->voteID)) {
                $vote_info["headline"] = $question_line[1];
                // The corresponding answers are in the same row ($i)
                // of the answer_file.
                $answer_line = explode("|", $answer_file[$i]);
                $vote_info["totalVotes"] = $answer_line[1];
                for ($j = 2; $j < sizeof($question_line) - 1; $j++) {
                    $vote_info["choice"][$j - 2] = $question_line[$j];
                    $vote_info["answer_absolute"][$j] = $answer_line[$j];
                    if ($vote_info["totalVotes"] == 0)
                        $vote_info["answer_relative"][$j - 2] = 0;
                else $vote_info["answer_relative"][$j - 2] = round(($vote_
info ["answer_absolute"][$j]
                        / $vote_info["totalVotes"]) * 100, 2);
                }
            }
        }
```

```
            return $vote_info;
        }
        //显示投票选项
        function _displayVote() {
            $vote_info = $this->_getVoteInfo();
            //调用显示选项的模板
            include ($this->choice_template);
            return $vote_info;
        }
        //该函数根据用户投票结果，修改保存投票信息的数组，然后将数据保存到文件中
        function _processVote(&$vote_info, $radiobutton, $voteOnce) {
            if ($this->_checkPermission($voteOnce) == 1) {
                $vote_info["totalVotes"]++;
                $vote_info["answer_absolute"][$radiobutton + 2]++;
                $vote_info["answer_relative"][$radiobutton] = round(($vote_info
["answer_absolute"][$radiobutton + 2]
                    / $vote_info["totalVotes"]) * 100, 2);
                $answer_file = file($this->answer_data);
                for ($i = 0; $i < sizeof($answer_file); $i++) {
                    $answer_line = explode("|", $answer_file[$i]);
                    // Find the selected poll in the DAT file.
                    if (!strcmp($answer_line[0], $this->voteID)) {
                        // Register the casted vote.
    $answer_line[$radiobutton+2]=$vote_info["answer_absolute"][$radiobutton + 2];
                        // Register the new total number of votes.
                        $answer_line[1] = $vote_info["totalVotes"];
                        $answer_file[$i] = implode("|", $answer_line);
                        $logged = $this->_writeArray($this->answer_data, $answer
_file, "w");
                    }
                }
            } else echo("对不起，你只能投票 1 次!<br><br>");
            return 0;
        }
        //显示投票结果
        function _displayResult($vote_info) {
            //计算每个投票选项的显示宽度
            for ($i = 0; $i < sizeof($vote_info["answer_relative"]); $i++)
                $width[$i] = 2 * round($vote_info["answer_relative"][$i]);
            // 调用显示投票结果的模板
            include($this->result_template);
        }
        //检查用户是否有投票权限，如果已经投票过，则取消投票权限
        function _checkPermission($voteOnce) {
            $votePermission = 1;
            // Check if the user has already voted for the actual poll.
            for ($i = 0; $i < sizeof($voteOnce); $i++) {
                if ((isset($voteOnce)) && ($voteOnce[$i] == $this->voteID)) {
                    // User has already voted.
                    $votePermission = 0;
                }
            }
```

```
            return $votePermission;
        }
        //保证每个用户只有一次投票机会，但是客户端必须支持 cookie
        function _setPermission($voteOnce) {
            $voteCookieIndex = sizeof($voteOnce);
            $setNewCookie = 1;
            for ($i = 0; $i < $voteCookieIndex; $i++) {
                if ($voteOnce[$i] == $this->voteID)
                    $setNewCookie = 0;
            }
            if ($setNewCookie == 1)
                header ("Set-Cookie: voteOnce[$voteCookieIndex]=$this->voteID;
                expires=Friday, 16-Jan-2037 00:00:00 GMT; path=/");
                return 0;
        }
        //该函数添加一个新的投票主题，依次按照文件格式写入对应的编号、主题和选项
        function _setNewVote($vote_info) {
            $question_file = file($this->question_data);
            $index = sizeof($question_file);
            $newQuestion[0] = $index;
            $newQuestion[1] = $vote_info["headline"];
            $newAnswer[0] = $index;
            $newAnswer[1] = 0;
            // 将选项写入文件
            // 初始化每个选项的投票结果为 0
            for ($i = 1; $i <= sizeof($vote_info["choice"]); $i++) {
                if ($vote_info["choice"][$i] != "") {
                    $newQuestion[$i + 1] = $vote_info["choice"][$i];
                    $newAnswer[$i + 1] = 0;
                }
            }
            $newQuestion = implode("|", $newQuestion)."|\r\n";;
            $newAnswer = implode("|", $newAnswer)."|\r\n";;
            $this->_writeLine($this->question_data, $newQuestion, "a");
            $this->_writeLine($this->answer_data, $newAnswer, "a");
            return 0;
        }
        //定义下拉菜单，让用户选择投票主题
        function _defineJumpMenu() {
            echo("<script language=\"JavaScript\" type=\"text/javascript\" >\n");
            echo("<!--\n");
            echo("function JumpMenu(targ,selObj,restore){\n");
    echo("eval(targ+\".location='\"+selObj.options[selObj.selectedIndex]. value
+\"'\")\n");
            echo("  if (restore) selObj.selectedIndex=0\n");
            echo("}\n");
            echo("//-->\n");
            echo("</script>\n");
            return 0;
        }
        //显示投票主题 ，让用户选择
        function _displayJumpMenu() {
```

```
            echo("<select class=\"selectVoteMenu\" name=\"VoteMenu\" onChange=
\"JumpMenu('parent',this,0)\">");
            echo("  <option selected>Select poll...</option>");
            $question_file = file($this->question_data);
            for ($i = sizeof($question_file) - 1; $i >= 0; $i--) {
                $question_line = explode("|", $question_file[$i]);
                echo("<option value=\"$this->pageName?ID=$question_line[0]\">");
                echo($question_line[1]);
                echo("</option>");
            }
            echo("</select>");
            return 0;
        }
    }
    ?>
```

3．实现管理员功能

投票系统的管理员功能由 admin.php 来实现。该文件比较简单，主要是显示表单控件，然后让用户输入投票主题以及投票选项。admin.php 文件如代码 14-9 所示。

【代码 14-9】 admin.php

```
    …省略部分 HTML 代码
    <?php
    if (isset($newVote)) {  //是否为数据提交
        include("class_support.php");
        include("class_vote.php");
        $vote_info["headline"] = $headline;
        for ($i = 1; $i < sizeof($choice); $i++)
            $vote_info["choice"][$i] = $choice[$i];
        $vote = new TSvote();  //创建类
        $vote->_setNewVote($vote_info);  //保存数据
    }
    ?>
```

管理员功能界面如图 14-11 所示。

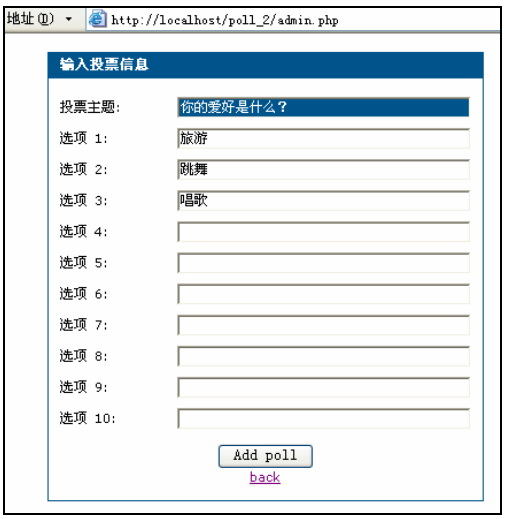


图 14-11 管理员功能界面

4．实现客户投票功能

客户投票功能主要在 demo.php 文件中实现。该文件中调用了类 Tsvote 来显示投票选项和投票结果。而 Tsvote 类调用了两个模板来显示，其中显示选项的模板文件为 choice_template.php，如代码 14-10 所示。

【代码 14-10】　choice_template.php

```
    <form name="vote" method="post" action="<?php echo($this->pageName); ?>">
      <table cellspacing="0" class="tableVoteChoice">
        <tr>
          <td class="tdVoteHeadline" colspan="2"><?php echo($vote_info["head-
line"]); ?></td>
        </tr>
        <?php
          //循环显示每个选项
          for ($i = 0; $i < sizeof($vote_info["choice"]); $i++) {
            ?>
            <tr>
              <td class="tdVoteChoiceLeft"><input type="radio" name="radiobutton"
value="<?php echo($i); ?>"></td>
              <td class="tdVoteChoiceRight"><?php echo($vote_info["choice"][$i])
 ?></td>
            </tr>
            <?php
          }
          ?>
        <tr>
          <td class="tdVoteChoiceBottom" colspan="2">
            <input type="submit" name="processVote" value="Vote!" class="buttonVote">
            <br>
            <br>
            <input type="hidden" name="ID" value="<?php echo($this->voteID); ?>">
            <a href="<?php echo("$this->pageName?action=displayResult&ID=".$this
->voteID); ?>">
              显示投票结果
            </a>
          </td>
        </tr>
      </table>
    </form>
```

显示投票结果的界面由模板文件 result_template.php 实现，如代码 14-11 所示。

【代码 14-11】　result_template.php

```
    <table cellspacing="0" class="tableVoteResult">
      <tr>
    //显示投票主题
     <td colspan="3" class="tdVoteHeadline"><?php echo($vote_info ["headline"]);
 ?></td>
      </tr>
      <?php
```

```
    //循环显示每个投票选项的结果
     for ($i = 0; $i < sizeof($vote_info["choice"]); $i++) {
       ?>
       <tr>
         <td class="tdVoteResultLeft"><?php echo($vote_info["choice"][$i]);
?></td>
         <!—图形化显示投票结果，是调用前面创建的类的方法来实现的-->
         <td class="tdVoteResultMiddle">
           <img src="<?php echo($this->bar_images[$i]); ?>" height="10" width=
"<?php echo($width[$i]); ?>" alt="bar">
         </td>
         <td class="tdVoteResultRight"><?php echo($vote_info["answer_relative"]
[$i]); ?> %</td>
       </tr>
       <?php
     }
     ?>
     <tr>
       <td colspan="3" class="tdVoteResultBottom"> 总投票数: <?php echo($vote_info
["totalVotes"]); ?></td>
     </tr>
   </table>
```

提示 由于 demo.php 文件比较简单这里省略其代码。

当用户在浏览器中输入“http://localhost/poll_2/demo.php”后，可以进行投票调查，界面如图 14-12 所示。

选择一个投票主题后，则会显示投票选项，如图 14-13 所示。

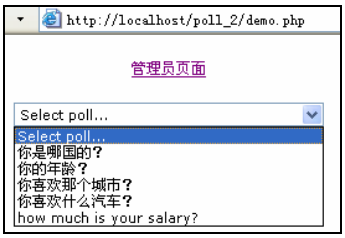


图 14-12 选择投票主题

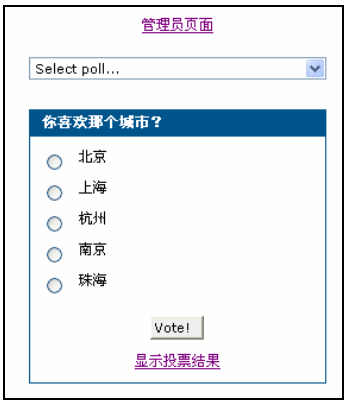


图 14-13 显示投票选项

当用户选择一个投票选项后，单击“Vote!”按钮，可以提交并查看提交结果，如图 14-14 所示。如果用户已经投票过一次，则会显示提示信息。

本系统还有一个功能就是可以限制投票次数，是通过 Cookie 来实现的，当用户投票后，需要向浏览器发送一个 Cookie 头，并且保存在客户端，该语句如下：

```
    header ("Set-Cookie: voteOnce[0]=$ID; expires=Friday, 16-Jan-2037 00:00:00
GMT; path=/");
```

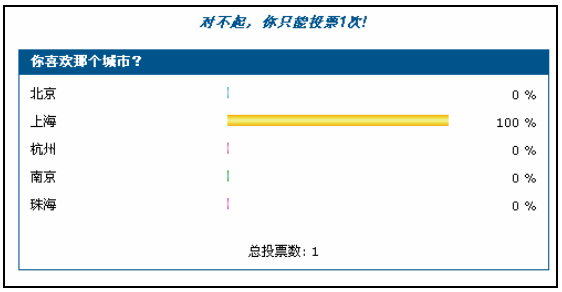


图 14-14　显示投票结果



要使得该功能发挥作用，客户端必须启用 Cookie，如果客户端浏览器禁止了 Cookie，则通过该方法就不能限制客户投票次数了。

第 15 章　计数器系统

使用计数器，网站可以统计每天访问的流量，也可以统计不同网页被浏览的次数。通常进行计数器设计时，可以采用数据库来保存统计结果，也可以通过文件来保存统计结果。

一个完整的计数器一般具备如下功能：

（1）统计当天的访问数量；

（2）统计所有的访问量。

本章将介绍 3 个计数器系统，分别是基于文本的图形显示计数器、面向对象的图形显示计数器和一个综合的计数器。

15.1　基于文本的图形显示计数器

该计数器的实现非常简单，只能简单地统计某个页面的总访问数量。每当新用户连接该页面或者刷新该页面的时候，计数器就会增加 1。统计数据被保存在一个文本文件中。

系统由 2 个文件组成。

（1）index.php，当用户连接该页面时，程序自动从文本文件中读出之前总的访问量，然后将数量增加 1，并保存文本文件。

（2）counter.txt，文本文件，用来保存统计数据。

【代码 15-1】　index.php

```
<?php
    //选择显示统计数据的颜色
    //$color_name="black_white";
    $color_name="white_black";
    //$color_name="black_transparent";
    // 只读方式打开文件
    $fp=fopen("counter.txt","r");
    //读取数据
    $counter=fgets($fp,1024);
    //关闭文件
    fclose($fp);
    //计数器增加 1
    $counter++;
    //可写方式打开文件
    $fp=fopen("counter.txt","w");
    //将新的统计数据写入文本文件
    fputs($fp,$counter);
    //关闭文件
    fclose($fp);
```

```
        //为了防止其他用户此时也访问了该页面，文件内容被改变
        //重新打开文件读取最新统计数据
        $fp=fopen("counter.txt","r");
        $counter=fgets($fp,1024);
        fclose($fp);
        //循环将统计数据用图像显示出来
        //不同的数字针对不同的图像
        //如数字 1，则用图像 1.gif 来显示
        for ($i=0;$i<strlen($counter);$i++)
        {
            $result=$counter[$i];
            switch($result)
            {
                case "0": $ret[$i]="0.gif"; break;
                case "1": $ret[$i]="1.gif"; break;
                case "2": $ret[$i]="2.gif"; break;
                case "3": $ret[$i]="3.gif"; break;
                case "4": $ret[$i]="4.gif"; break;
                case "5": $ret[$i]="5.gif"; break;
                case "6": $ret[$i]="6.gif"; break;
                case "7": $ret[$i]="7.gif"; break;
                case "8": $ret[$i]="8.gif"; break;
                case "9": $ret[$i]="9.gif"; break;
            }
        }
        echo "该页面的总访问次数为：";
        //循环输出图像
        for ($i=0;$i<sizeof($ret);$i++)
            echo "<img border=\"0\" src=\"$color_name/$ret[$i]\" width=\"8\"
height=\"11\">";
    ?>
```

在代码 15-1 中，变量$color_name 代表了图像的路径，采用不同的名称，则使用不同的图像来表示。系统提供了 3 种图形来显示数字，读者可以根据需要来选择。

计数器系统目录为“counter_1”，将其拷贝至服务器根目录下，然后在浏览器中输入“http://localhost/counter_1/index.php”，即可显示结果，执行结果如图 15-1 所示。

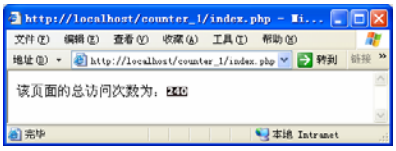


图 15-1　计数器显示结果

15.2　面向对象的图形显示计数器

下面介绍的计数器采用面向对象的方法实现的。它也支持通过数据库来保存统计数据。

根据对计数器功能的分析，系统由以下文件组成。

（1）counter.php，用于调用计数器的实现类 mCounter，并显示计数结果。

（2）counter.inc，定义了一个实现类，该类中定义了很多接口函数，可以读取文本文件的

数据，也支持数据库存储。

（3）databases.inc，该文件声明了一个数据库操作的存储类。

（4）default.html，需要被统计的任何网页。

15.2.1 counter.php

counter.php 文件代码非常简单，直接创建一个类，然后调用类方法，如代码 15-2 所示。

【代码 15-2】 counter.php

```
<?PHP
Require "databases.inc";
Require "counter.inc";
$c = New mCounter;
echo $c->ShowAsImage();
?>
```

15.2.2 counter.inc

该文件主要声明了一个类 mCounter，该类能够存储文本文件数据，也能够调用数据库类操作数据库，如代码 15-3 所示。

【代码 15-3】 counter.inc

```
<?php
Class mCounter
{
    // ---------------------------------------------------------------------
    //以下为类属性声明
    var $type; // 计数器保存类型，f (默认为文件保存) , d 表示数据库
    // 初始化计数器的值
    var $startValue;
    // 文件名称
    var $counterFile;
    // 计数器中的位置
    var $positions;
    // 显示计数的 x 坐标
    var $xStart;
    //显示计数的 y 坐标
    var $yStart;
    // 数字宽度
    var $widthStart;
    // 数字高度
    var $heightStart;
    // 数字间距
    var $xInterval;
    // 计数器显示的页面名称
    var $frameName;
    // 计数器显示位置
    var $imagesDir;
    Function mCounter() {  //初始化变量值
```

```
            // 数据库变量
            $this->dbtype = "pg";
            $this->connecttype = "c";
            $this->connect = "";
            $this->username = "";
            $this->password = "";
            $this->dbname = "";
            $this->query = "";
            // 计数器变量
            $this->type = "f";
            $this->startValue = 0;
            $this->counterFile = "../counter/counter.txt";
            $this->startValue = 0;
            $this->positions = 6;
            $this->xStart = 2;
            $this->yStart = 2;
            $this->widthStart = 15;
            $this->heightStart = 20;
            $this->xInterval = 1;
            $this->frameName = "../images/counter.jpg";
            $this->imagesDir = "../images/";
        }
        Function Number($value, $position) {  //获取计数器的数字
            $strVal = $value;
            $length = StrLen($strVal);
            If ($length < $this->positions) {
                $strVal = Str_Repeat("0", ($this->positions - $length)) . $strVal;
            }
            Return Substr($strVal, $position, 1);
        }
        Function ShowAsImage() {  //显示计数器图像
            Switch ($this->type) {
                Case "f":
            $value = $this->FileNewValue();
            Break;
            Case "d":
            $value = $this->DBNewValue();
            Break;
            }
            Header ("Content-type: image/jpeg");
            $frame = ImageCreateFromJPEG($this->frameName);
            $strVal = $value;
            $length = StrLen($strVal);
            If ($length < $this->positions) {
                $strVal = Str_Repeat("0", ($this->positions - $length)) . $strVal;
            }
            For ($i = 0; $i < $this->positions; $i++) {
                If (($this->positions - $length) > $i) {
                    $file = $this->imagesDir . "0a.jpg";
                } Else {
                    $file = $this->imagesDir . Substr($strVal, $i, 1) . ".jpg";
                }
```

```
            $img = ImageCreateFromJPEG($file);
            $r = ImageCopy($frame, $img, $this->xStart, $this->yStart, 0, 0,
                $this->widthStart, $this->heightStart);
            $this->xStart = $this->xStart + $this->widthStart + $this->xInterval;
        }
        Return ImageJPEG($frame);
    }
    Function ShowAsText() {
        Switch ($this->type) {
            Case "f":
        Return $this->FileNewValue();
        Case "d":
        Return $this->DBNewValue();
        }
    }
    Function FileNewValue() {  //将计数器增加 1 并保存进文件
        if (!file_exists($this->counterFile)) {
            Return 0;
        }
        $fp = fOpen($this->counterFile, "r+");
        $count = (int) fGets($fp, $this->positions + 1);
        $max =  (int) Str_Repeat("9", $this->positions);
        If ($count == $max) {
            $count = 1;
        } Else {
            $count += 1;
            $strCount = $count;
        }
        fTruncate($fp, 0);
        fSeek($fp, 0);
        fWrite($fp, $count, $this->positions + 1);
        fClose($fp);
        Return $count;
    }
    Function DBNewValue() {  //将计数器增加 1 并保存到数据库
        // this part use another class: see "DataBases.inc"
        $dbObj = New mDatabase;
        $c = $dbObj->Open($this->dbtype, $this->connecttype, $this->connect,
            $this->username, $this->password);
        If ($this->dbtype != "pg") $dbObj->SelectDB($this->dbname);
        $query = "UPDATE Counter SET idNumber = NewNumber" .
            " FROM Counter," .
            " (SELECT (MAX(idNumber) + 1) AS NewNumber" .
            " FROM Counter" .
            " ) AS alsMax";
        $rUpdate = $dbObj->Query($query);
        If (!$rUpdate) Return 0;
        $query = "SELECT idNumber FROM Counter";
        $rSelect = $dbObj->Query($query);
        If (!$rSelect) Return 0;
        $d = $dbObj->MoveFirstRec($rSelect);
        $count = $d[0];
```

```
            $dbObj->Close;
            Return $count;
        }
    }
```

15.2.3　databases.inc

该文件定义了一个数据库操作类，该类具有很好的移植性，并且能够操作多种数据库。读者只要掌握了该类，基本的数据库操作就会游刃有余了。

该类中定义了以下重要函数。

Open：打开数据库连接。

Close：关闭数据库连接。

SelectDB：选择数据库。

Query：创建查询。

DataSeek：移动记录指针。

FieldName：获取字段名称。

FieldType：获取字段类型。

FieldLenght：获取字段长度。

FetchRow：获取数据。

FetchArray：获取数据并保存进数组。

FetchObject：获取数据并保存到对象。

Result：获取结果数据。

FreeResult：刷新记录集。

RowsNumber：获取记录数量。

FieldsNumber：获取字段数量。

CurRecNumber：获取当前记录行号（从 0 开始）。

MoveFirstRec：移到第一条记录。

MoveLastRec：移到最后一条记录。

MovePreviousRec：移到前一条记录。

MoveNextRec：移到下一条记录。

MoveToRec：移动到指定记录。

databases.inc 文件代码较长，由于篇幅所限，此处省略，读者可对照光盘中提供的代码学习。

15.2.4　实例运行演示

为了方便演示，笔者将计数器数据保存在文件中，保存计数器的文件是“counter/counter.txt”。用户在浏览器中输入“http://localhost/counter_2/default.html”，显示的结果如图 15-2 所示。



图 15-2　显示统计结果

其中 default.html 文件是需要被统计访问次数的页面，如代码 15-4 所示。

【代码 15-4】 default.html

```
<HTML>
<HEAD>
<BODY>
总访问人数为：
<P><IMG SRC="lib\counter.php" WIDTH="100" HEIGHT="24" ALT="forway counter"></P>
</BODY>
</HTML>
```

15.3 综合计数器

接下来介绍一个综合性的计数器。该计数器能够实现如下功能：

（1）统计每个页面的当天访问次数；

（2）总访问次数；

（3）显示一个日历，让用户随便查看任何一天的访问流量；

（4）记录用户每次访问页面的时间。

该系统由以下文件组成：

（1）counter.php，用于显示日历以及统计结果；

（2）dbconnection.php，用于连接数据库；

（3）top.php，用于导航，将该文件放置在需要统计流量的网页中，即可进行统计；

（4）counterdb.sql，该文件为创建数据库时所用的 SQL 语句。

15.3.1 配置数据库

本系统采用数据库保存统计结果，因此首先要创建数据库，数据库名称为“counterdb”，然后执行以下 SQL 语句创建数据库表 mycount：

```
CREATE TABLE mycount (
  mycntin int(4) NOT NULL auto_increment,
  mycnt int(4) DEFAULT '0' NOT NULL,
  mytime varchar(20),
  PRIMARY KEY (mycntin)
);
```

提示 该 SQL 语句保存在 counterdb.sql 文件中。

15.3.2 显示日历及统计结果（counter.php）

该文件是整个计数器的核心文件。首先生成一个日历，用户可以随意查看每个月的日期，单击某个具体日期可以查看当日的流量统计。

counter.php 文件如代码 15-5 所示。

【代码 15-5】 counter.php

```
<?php
    //检测是否声明了$ connect 变量
```

```
    if (empty($connect) OR $connect!='1')
    {
            $connect = include ("dbconnection.php");
    }
    $db = new classname;  //创建类实例
    $db->init();
    if (!$db->init())  //数据库初始化
    {
        echo "Connection Failed";
    }
    //获取提交的年、月、日
    if (isset($_GET['year']))
        $year=$_GET['year'];
    else
        $year="";

    if (isset($_GET['month']))
        $month=$_GET['month'];
    else
        $month="";
    if (isset($_GET['day']))
        $day=$_GET['day'];
    else
        $day="";

    //是否选择日期
    if(isset($_GET['fromcounter']))
    {   //如果选择了，则根据提交的年、月、日创建时间戳
        $todaytime= mktime (0,0,0,$month,$day,$year);
        $dtodaytime =date("d",$todaytime);
        $mtodaytime =date("m",$todaytime);
        $Ytodaytime =date("Y",$todaytime);
    }
    else
    {   //否则利用当前时间
        $todaytime=time();
        $dtodaytime =date("d",$todaytime);
        $mtodaytime =date("m",$todaytime);
        $Ytodaytime =date("Y",$todaytime);
    }
    //End Connection
class Calendar  //创建一个日历类
{
    function Calendar()
    {
    }
    function getDayNames()  //获取日期
    {
       return $this->dayNames;
    }
    function setDayNames($names)  //设置日期
    {
```

```
        $this->dayNames = $names;
    }
    function getMonthNames()  //获取月份名称
    {
        return $this->monthNames;
    }
    function setMonthNames($names)  //设置月份名称
    {
        $this->monthNames = $names;
    }
    function getStartDay()   //获取开始日期
    {
        return $this->startDay;
    }
    function setStartDay($day)   //设置开始日期
    {
        $this->startDay = $day;
    }
    function getStartMonth()  //获取开始月份
    {
        return $this->startMonth;
    }
    function setStartMonth($month)  //设置开始月份
    {
        $this->startMonth = $month;
    }
    function getCalendarLink($month, $year)
    {
        return "";
    }
    function getDateLink($day, $month, $year) //获取日期链接
    {
     $link = "counter.php?day=$day&month=$month&year= $year&fromcounter=1";
        return "$link";
    }
    function getCurrentMonthView()  //设置日期链接
    {
        $d = getdate(time());
        return $this->getMonthView($d["mon"], $d["year"]);
    }
    function getCurrentYearView()  //获取当前年份
    {
        $d = getdate(time());
        return $this->getYearView($d["year"]);
    }
    function getMonthView($month, $year)  //获取指定年、月日历
    {
        return $this->getMonthHTML($month, $year);
    }
    function getYearView($year)  //获取指定年的月份信息
    {
        return $this->getYearHTML($year);
```

```
}
function getDaysInMonth($month, $year)  //获取每个月的天数
{
   if ($month < 1 || $month > 12)
   {
      return 0;
   }
   $d = $this->daysInMonth[$month - 1];
   if ($month == 2)
   {
      if ($year%4 == 0)
      {
         if ($year%100 == 0)
         {
            if ($year%400 == 0)
            {
               $d = 29;
            }
         }
         else
         {
            $d = 29;
         }
      }
   }
   return $d;
}
function getMonthHTML($m, $y, $showYear = 1)
{ //根据月份显示日期信息
    //省略部分代码
}
function getYearHTML($year)
{  //根据年份显示月份信息
    //省略部分代码
}
//调整日期
function adjustDate($month, $year)
{
   $a = array();
   $a[0] = $month;   //月份
   $a[1] = $year;  //年份
   while ($a[0] > 12)  //如果月份大于 12
   {
      $a[0] -= 12;   //月份减 12
      $a[1]++;  //年份增加 1
   }
   while ($a[0] <= 0)  //如果月份小于 0
   {
      $a[0] += 12;   //月份加 12
      $a[1]--;   //年份减 1
   }
   return $a;
```

```
        }
         //输出相关 html 信息
    }
    class MyCalendar extends Calendar
    {  //创建一个子类，从 Calendar 类继承而来
        function getCalendarLink($month, $year)
        {
            $s = getenv('SCRIPT_NAME');
            return "$s?month=$month&year=$year";
        }
    }
    $d = getdate(time());  //获取时间
    if ($month == "")
    {
        $month = $d["mon"];  //获取月份
    }
    if ($year == "")
    {
        $year = $d["year"];  //获取年份
    }
    $cal = new MyCalendar;  //创建日历类实例
    echo $cal->getMonthView($month, $year);
    print("<br>");
    //查询出计数值
    $qry="SELECT * FROM mycount";
    $result=mysql_query($qry);
    $num=mysql_num_rows($result);
    $count=0;
    echo "
         <table border=4 cellspacing=4 cellpadding=4 align=center bordercolor=maroon>
             <tr>
                 <td><font face='courier new' color='red'>访问编号</td><td><font
face='courier new' color='red'>日期</td><td><font face='courier new' color='red'>
时间</td>
             </tr> ";
    if($row=mysql_fetch_array($result))
    {//显示统计结果
         do
         {
             $mycnt=$row['mycnt'];
             $mytime=$row['mytime'];
             $dshowtime = date("d",$mytime);
             $mshowtime = date("m",$mytime);
             $Yshowtime = date("Y",$mytime);
             $showtimenow1 = date("D dS, F  Y ",$mytime);
             $showtimenow2 = date("g:i a",$mytime);
             if(($dshowtime==$dtodaytime) AND ($mshowtime==$mtodaytime) AND
($Yshowtime==$Ytodaytime))
             {
                 echo "
                 <tr>
                     <td><font face='courier new' color='blue'>Hit Number $mycnt</td>
```

```
                    <td><font face='courier new' color='blue'>$showtimenow1</td>
                    <td><font face='courier new' color='blue'>$showtimenow2</td>
                </tr>
                ";
                $count++;
            }

        }
        while($row=mysql_fetch_array($result));
    }
    echo "
            <tr>
                <td></td><td><font face='courier new' color='green'>今日访问量:
<b>$count</b></td>
            </tr>
            <tr>
                <td></td><td><font face='courier new' color='green'>总访问量 :
<b>$num</b></td>
            </tr>
        </table>
    ";
    ?>
```

15.3.3　连接数据库（dbconnection.php）

该文件用于连接负责计数的后台 MySQL 数据库，文件比较简单，主要是进行 MySQL 常用函数的调用。

【代码 15-6】　dbconnection.php

```
    <?php
    class classname
    {
        // 数据库连接属性
        var $CONN = "";
        var $IPADD = "localhost";  //服务器地址，也可以使用 IP 地址
        var $USER = "root"; //database username
        var $PASS = "123456";       //database password
        var $DBASE = "counterdb";      //database name
        //End Connection Variables
        function error($text)
        {
            $no = mysql_errno ();
            $msg = mysql_error ();
            echo "[$text] ( $no : $msg) <br> \n";
            exit ;
        }
        function init()  //初始化数据库连接
        {
            $ipadd = $this->IPADD;
            $user = $this->USER;
            $pass = $this->PASS;
```

```
            $dbase = $this->DBASE;
            $conn = mysql_pconnect ($ipadd,$user,$pass);
            if (!$conn)
            {
                $this->error ("Connection could not be established with server");
            }
            if (!mysql_select_db($dbase))
            {
                $this->error ("Connention with database failed");
            }
            $this->CONN = $conn;
            return true;
        }
    }
```

15.3.4 导航文件（top.php）

该文件用于显示每个页面的被访问的人数以及当前的时间。使用时，可以将该文件放在任何网页的顶部或者其他位置，当用户进入该网页时，会看到流量统计，如代码 15-7 所示。

【代码 15-7】 top.php

```
<?php
    if (empty($connect) OR $connect!='1')
    {
            $connect = include ("dbconnection.php");
    }
    $db = new classname;
    $db->init();
    if (!$db->init())
    {
        echo "Connection Failed";
    }
    //End Connection
    //查询显示总访问量
    $qry0 = "SELECT max(mycnt) FROM mycount";
    $res0 = mysql_query($qry0);
    if($row=mysql_fetch_array($res0))
    {
        do
        {
            $newcnt=$row['max(mycnt)'];
        }
        while($row=mysql_fetch_array($res0));
    }
    //总访问量增加 1，并写入数据库，同时写入访问时间
    $newcnt=$newcnt+1;
    $mytime=time();
    $qry = "INSERT INTO mycount (mycnt,mytime) VALUES ($newcnt,'$mytime')";
    $res1 = mysql_query($qry);
    $showtime =date("D dS, F  Y , g:i a",$mytime);
    $email = "james_smith73@yahoo.com";
```

```
        $to = "forway73@yahoo.com";
        $from = "forway Counter";
        $subject = "Hit Registered";
        $msg = "";
        $msg = "<font face=arial size=2>You had a visitor to ur site at $showtime
</font><br><br> \n";
       $Fromaddress="james_smith73@yahoo.com";
        //mail($to." <".$email.">", $subject, $msg, "From: ".$from."
<".$Fromaddress.">\nContent-Type: text/html; charset=iso-8859-1");
        print("<table border=1 align='center' width=100% ><tr ><td  align=left><font
face='Arial'>   从 2005 年 12 月 19 日开始访问人数是: <b>$newcnt</b> </font></td>
<td align=right><font face='Arial' >当前时间 :: $showtime ; </td></tr></table>");
    ?>
    </body>
    </html>
    <textarea rows="1" cols="1" style="display:none;">
```

15.3.5　实例运行演示

下面来运行该综合计数器系统，比如直接连接“top.php”文件，显示结果如图 15-3 所示。



图 15-3　显示统计结果

如果直接进入“counter.php”文件，可以精确查看每日的访问统计以及综合统计数据，同时可以统计出每个用户访问的时间，如图 15-4 所示。



<<	November 2005					>>
Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

访问编号	日期	时间
Hit Number 1	Tue 22nd, November 2005	12:01 am
Hit Number 2	Tue 22nd, November 2005	12:02 am
Hit Number 3	Tue 22nd, November 2005	12:02 am
Hit Number 4	Tue 22nd, November 2005	12:02 am
Hit Number 5	Tue 22nd, November 2005	12:02 am
Hit Number 6	Tue 22nd, November 2005	12:02 am
Hit Number 7	Tue 22nd, November 2005	12:02 am
	今日访问量 : 7	
	总访问量 : 19	

图 15-4

至此，本章一共介绍了 3 种计数器的实现。读者可以根据自己的需要以及实际掌握的 PHP 知识在自己的网站建设中选择不同的计数器。

第 16 章　文件管理系统

本章将通过一个综合的实例介绍如何进行文件的各种操作。

16.1　系统整体设计

根据文件管理系统的功能需求，设计本系统将包含以下几个文件。

（1）index.php，为导航文件，当用户在浏览器中输入该地址，则自动链接到 explorer.php 文件。

（2）explorer.php，整个系统的主文件，用于显示指定路径的文件夹以及文件和相关信息。

（3）mkdir.php，用于创建一个指定的文件夹。

（4）rmdir.php，用于删除一个指定的文件夹。

（5）save.php，用于保存一个被编辑过的文件。

（6）send.php，用于发送到浏览器显示某个文件。

（7）delfile.php，用于提示删除一个文件。

（8）dodelfile.php，用于删除一个文件。

（9）edit.php，用于编辑一个文件。

（10）expl2save.php，用于将文件上传到服务器。

（11）unreadf.php，用于显示错误提示。

16.2　系 统 实 现

接下来将分别介绍重要文件的核心代码实现。

16.2.1　浏览目录和文件的基本信息

explorer.php 文件是系统的主文件，用于显示文件列表以及文件属性。当用户在浏览器中输入“http://localhost/file/index.php”便会自动导航到“http://localhost/file/explorer.php”，因为 index.php 文件非常简单，只是实现一个重定向的功能，代码如下所示：

```
<?php Header("Location: explorer.php") ?>
```

当链接到 explorer.php，浏览器显示的界面如图 16-1 所示。

单击任何一个文件夹，可以查看其子目录和子文件夹，如图 16-2 所示。

在图 16-2 中单击“”图标能够回到上一级目录。该系统能够根据不同的文件类型显示不同的图标。另外也可以显示文件不同的属性，比如为只读、读写等。

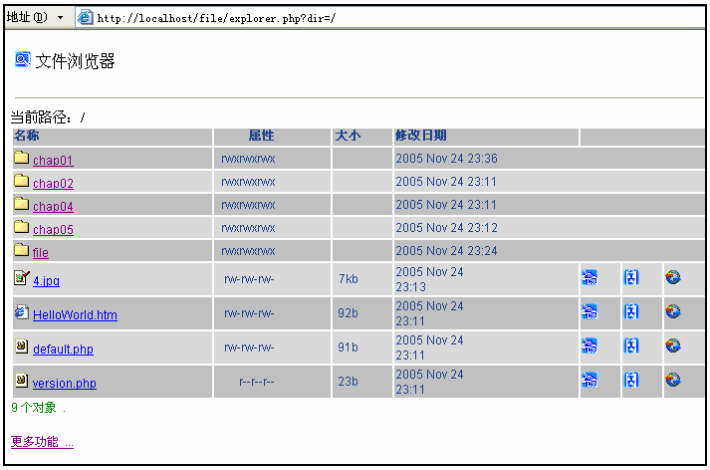


图 16-1　文件管理系统的主界面



图 16-2　查看子文件夹和子文件

explorer.php 文件如代码 16-1 所示。

【代码 16-1】　explorer.php

```
    <?php
    require("path.ini");  //包含 path.ini 文件
    //设定常量
    define("SHOW_EDIT",1);   //显示编辑按钮
    define("HIDE_EDIT",0);  //隐藏编辑按钮
    /*检查访问权限*/
    require("explorer.cfg");
    /* 获取要浏览的目录路径*/
    if (isset($_GET["dir"]))
        $dir=$_GET["dir"];
    /*检查路径安全权限*/
    require("lib/inc/security.inc");
    //检查是否允许浏览
    $allowbrowse= $AllowSelfBrowse || !strstr($dir,basename(dirname($GLOBALS
["SCRIPT_FILENAME"])));
    //if(isset($IsCheckAccess)) {  //是否需要
        //include("access.php");
    //}
    /*包含检查文件类型的文件 mime.inc*/
    require("lib/inc/mime.inc");
    /***********************************************/
    function GetFileInfo($f) {  //获取文件信息
```

```
$s = array (    'perm'  => '',
        'humanperm' => '',  //权限
        'ux'    => false,   //是否为可执行文件
        'islnk' => false,   //是否为一个链接
        'owner'  => '',  //所有者
        'group'  => '',  //文件组
        'size'   => 0,  //大小
        'mtime'  => ''  //文件类型
        );
$mystat=stat($f);
//检查权限
$s["perm"]=$mystat[2] & 511;
if(($s["perm"] & 320) == 320) $s['ux']=true;
/*将权限改成可识别模式 */
$fullperms=256; // rwxrwxrwx
$s["humanperm"]="";
$let = array('r','x','w'); //no, it's not rwx, but rxw, because browsed backwards

for($i=9;$i>0;$i--) {
    ($s["perm"] & $fullperms)? $s["humanperm"].=$let[$i % 3]:$s["humanperm"].="-";
    $fullperms = $fullperms >> 1;
}
//判断是否是一个链接
if(!strcmp(filetype($f),"link")) $s['islnk']=true;
//文件所有者
$s["owner"]=$mystat[4];
//文件组
$s["group"]=$mystat[5];
//文件大小
$s["size"]=$mystat[7];
//文件最后修改时间
$s["mtime"]=date('Y M d H:s',$mystat[10]);
return($s);
}
/*
/*********************************************/
?>
<html>
<head>
<title>File Explorer - Tropicalm Admin</title>
<link rel=stylesheet href="lib/explorer.css" type="text/css">
<script language=JavaScript src="lib/more.js"></script>
</head>
<body bgcolor=white>
<table border=0 cellspacing=3 width=700>
  <tr>
    <td colspan=2>
     <div class=title1>
     <img src="images/explorer.gif" border=0>
     文件浏览器</div>
     <br>
     <hr size=1>
```

```
      </td>
    </tr>
  </table>
  <?php
  /*如果路径存在，则打开!*/
  $hDir=opendir($ROOTPATH.$dir);
  /*output the current path*/
  print "<div class=title2>\n";
  if($allowbrowse) {
      print "当前路径: ".$dir;
      if(strcmp($dir,'/')) print "/";
  }
  print "</div>\n";
  /*浏览目录*/
  $j=0;
  $frmid=0;
  //获取所有文件和目录
  //prepare the exceptions (file to hide)
  if(strlen($Exceptions)>0) $except=explode(",",$Exceptions);
  while ($file=readdir($hDir)) {
      $res=true;
      if(strlen($Exceptions)>0) {
          reset($except);
          for($i=0;$i<count($except);$i++) {
              if(!strcmp($file,$except[$i])) {
                  $res=false;
                  break;
              }
          }
      }
      //被显示的文件
      if($res) {
          $files[$j++]=$file;
      }
  } /*end while*/
  /*上一级目录*/
  if($allowbrowse) {
      if(strcmp($dir,"/"))   {
          $updir=dirname($dir);
          $updir=str_replace("\\","/",$updir);

          if($updir=="")
              $updir="/";
          print "<div class=normal><a href='explorer.php?dir=$updir'><img src=
'images/updir.gif' border=0 alt='updir' width=16 height=16></a></div>\n";
      }
      /*对文件排序*/
      if(count($files)) {
          sort($files);
          reset($files);
          /*用表格形式输出文件*/
          print "<table border=0 width=";
```

```
            ($IsShowDetails)? print "700" : print "288";
            print " class=normal>\n";
            /*打印标题*/
            print "  <tr>\n";
            print "   <td class=bdyResult0 width=80><b>名称</b></td>\n";
            //详细信息
            if($IsShowDetails) {
                print "   <td class=bdyResult0 width=60 height=10 align=right><b>
属性</b></td>\n";
                print "   <td class=bdyResult0 width=30 height=10 align=center><b>
大小</b></td>\n";
                print "   <td class=bdyResult0 width=80 height=10 align=left><b>修
改日期</b></td>\n";
            }
            print "   <td class=bdyResult0 width=48 colspan=3 height=10> 
</td>\n";
            print "  </tr>\n";
            //总对象数量，包括目录和文件
            $i=0;
            while (list($key,$name) = each($files)) {
                //选择显示样式
                $class= "bdyResult".($i%2);
                $filename=$ROOTPATH."$dir$sp$name";
                $rc=chkfile($name,$icon);
                //当前文件是否为文件夹
                if(is_dir($filename)) {
                    $filestats=GetFileInfo($filename);
                    if(!$filestats['ux'] || !$AllowSelfBrowse && !strcmp($name,
basename(dirname($GLOBALS["SCRIPT_FILENAME"]))) ) {
                        // 显示名称
                        print "  <tr>\n";
                        print "   <td class=$class width=80>\n";
                        print "\t<img border=0 src='images/icon_nodir.gif' alt=
'[private dir]' width=16 height=16> $name\n";
                        print "   </td>\n";
                    }
                    else {
                        // display name
                        print "  <tr>\n";
                        print "   <td class=$class width=80>\n";
                        $href="explorer.php?dir=$dir$sp$name";
                print "\t<a href=\"".$href."\"><img border=0 width=16 height=16 ";
                        if($filestats['islnk'])
                print "src='images/icon_dirlnk.gif' alt='[dir link]'></a>";
                        else
                            print "src='images/icon_dir.gif' alt='[dir]'></a>";
                        print " <a href=\"".$href."\">$name</a>\n";
                        print "   </td>\n";
                    }
                    //显示详细信息
                    if($IsShowDetails) {
                        //显示权限
```

```
                print "   <td class=$class width=60 height=10 align=right>";
                        print $filestats["humanperm"];
                        print "</td>\n";
                        //显示大小
                   print " <td class=$class width=30 height=10 align=right> </td>\n";
                        //显示修改日期
            print "    <td class=$class colspan=4 width=168 height=10 align=left>";
                        print $filestats["mtime"];
                        print "</td>\n";
                        print "  </tr>\n";
                        $i++;
                    }
                }
            }
            reset($files);
            while (list($key,$name) = each($files)) {

                $rc=chkfile($name,$icon);
                $filename=$ROOTPATH."$dir$sp$name";
                $href=$ROOTURL."$dir$sp$name";
                //如果为文件
                if(!is_dir($filename)) {
                    $class= "bdyResult".($i%2);
                    $filestats=GetFileInfo($filename);
                    print "  <tr>\n";
            print "   <td class=$class width=80 height=10 valign=middle>\n";
            print "\t<a href=\"".$href."\"><img border=0 src='$icon' alt='[file]'
width=16 height=16></a>";
                    print " <a href=\"".$href."\">$name</a>\n";
                    print "   </td>\n";
                    //显示详细信息
                    if($IsShowDetails) {
                        //display permissions
                print "   <td class=$class width=60 height=10 align=right>";
                        print $filestats["humanperm"];
                        print "</td>\n";
                        //显示大小
                        print "   <td class=$class width=30 height=10 align=center>";
                        $s=$filestats["size"];
                        $unit="b";
                        if($s >= 1024) {
                            $s=floor(round($s/1024));
                            $unit="kb";
                        }
                        if($s>=1024*1024) {
                            $s=floor(round($s/1048576));
                            $unit="Mb";
                        }
                        print $s.$unit;
                        print "</td>\n";
                        //显示修改日期
                        print "   <td class=$class width=80 height=10 align=left>";
```

```
                        print $filestats["mtime"];
                        print "</td>\n";
                    }
                    /*如果为文本文件，则显示可编辑的图标*/
                    if($rc) {
                        print "   <td class=$class valign=middle width=16>\n";
                        print "\t<a href='edit.php?dir=$dir&name=$name'><img src=
'images/edit.gif' border=0 alt='edit'></a>\n";
                        print "   </td>\n";
                    }
                    else {
                        print "   <td class=$class width=16> </td>\n";
                    }
                    //显示删除图标
                    if($IsShowDelete) {
                        print "   <td class=$class valign=middle width=16>\n";
                        print "\t<a href='delfile.php?dir=$dir&name=$name'><img
src='images/del.gif' border=0 alt='delete' width=16 height=16></a>\n";
                        print "   </td>\n";
                    }
                    else {
                        print "   <td class=$class width=16> </td>\n";
                    }
                    /*显示发送到浏览器用于显示的图标*/
                    print "   <td class=$class valign=middle width=16>\n";
                    print "\t<a href='send.php?filename=$dir$sp$name'><img src=
'images/xfer.gif' alt='send' border=0 width=16 height=16></a>\n";
                    print "\t   </td>\n";
                    print "  </tr>\n";
                $i++;
                }
            } //end of while
        print "</table>\n\n";
        //总对象数
        $msg=$i."   个对象 ";
        //if($i>1) $msg.="s";
        print "<div class=comment>$msg.</div>\n";
        } //end of if(count($files))
        else {
            print "<br><div class=normal>[Empty]</div>\n";
        }
        print "<div class=normal><p><a href=\"javascript:shwMoreOpt('moreopt.php?
dir=$dir',430,330,0)\">更多功能 ...</a></p></div>\n";
    } //endif allowbrowse
    else {
        if(! $pass++) {
        print "  <div class=error>browsing under the explorer itself is forbidden
(see your administrator)</div>\n";
        }
    }
    ?>
    <hr size=1 width=700 align=left>
```

```
    </body>
    </html>
```

以上文件是文件管理系统的主文件，首先检查该目录是否能够被浏览，如果可以，则循环读取所有的文件和目录。在显示文件的时候，需要判断文件的类型，如果文件可编辑，则显示编辑图标，如果该文件能够被浏览器浏览，则显示发送到浏览器的图标，另外还要显示删除图标，方便用户删除文件。

16.2.2　删除文件

既然是文件管理系统，那么删除文件功能是必不可少的。在图 16-1 所示的每个文件后面有 3 个图标，第 2 个图标“”表示删除文件，当用户单击该图标后，会进入到删除文件页面，如图 16-3 所示，该页面文件名为 delfile.php，如代码 16-2 所示。



图 16-3　删除文件页面

【代码 16-2】　delfile.php

```
    <?php
    //省略部分代码
    ?>
    </head>
    <body bgcolor=white>
    <table border=0 width=500 cellspacing=1 cellpadding=1>
    <form name=myForm>
      <tr>
        <td colspan=2 class=title1 >
         <img src="images/del.gif" border=0>
         文件删除确认：  <?php print $dir.$sp.$name ?>
         <br>
         <a href="explorer.php?dir=<?php print $dir.$sp.$name; ?>" class=href>
file explorer</a>
         <hr size=1 width=100%>
        </td>
      </tr>
      <tr>
        <td colspan=2 class=bdyForm>确认要删除文件<?php print $dir.$sp.$name ?>?吗? </td>
      </tr>
      <tr>
        <td align=left class=bdyForm>
         <input type=button class=combtn name="cmdNo" value="  No  " onClick="go
('explorer.php?dir=<?php print $dir; ?>')">
        </td>
        <td align=right class=bdyForm>
         <input type=button class=combtn value=" Yes " onclick="go('dodelfile.
php?<?php print "dir=$dir&name=$name" ;?>')" >
```

```
      </td>
    </tr>
  </form>
  </table>
  <?php
  //省略部分代码
  ?>
```

但是该页面并没有真正删除文件，只是让用户确认。只有用户单击“Yes”时，才能够正式将文件删除。而删除文件是在 dodelfile.php 中实现的，如代码 16-3 所示。文件删除后的界面如图 16-4 所示。



图 16-4　文件成功删除

【代码 16-3】　dodelfile.php

```
  <?php
  /*包含路径设置文件*/
  require("path.ini");
  /*检查访问权限*/
  require("explorer.cfg");
  /* 获取路径信息*/
  $dir=$_GET["dir"];
  $name=$_GET["name"];
  /*检查路径安全属性*/
  require("lib/inc/security.inc");
  //获取是否允许被浏览
  $allowbrowse= $AllowSelfBrowse || !strstr($dir,basename(dirname($GLOBALS
["SCRIPT_FILENAME"])));
  if(!$allowbrowse) {
      print "  <div >browsing under the explorer itself is forbidden (see your
administrator)</div>\n";
      exit();
  }
  /*************************************************/
  //如果文件存在同时该文件能够被删除
  if(file_exists($ROOTPATH.$dir.$sp.$name) && is_writeable($ROOTPATH.$dir.
$sp.$name)) {
      /*delete it*/
      if(!$IsSim)
          //调用 unlink 删除该文件
          $rc=unlink($ROOTPATH.$dir.$sp.$name);
  }
  else {
      Header("Location:
unreadf.php?filename=".urlencode($ROOTPATH.$dir.$sp.$name));
      exit();
  }
```

```
?>
<?php
//省略部分代码
?>
```

16.2.3　编辑文件

如果系统判断文件是一个文本文件，则提供一个用于编辑的图标“”，即图 16-1 所示中文件名后的第 1 个图标。单击此图标，便会进入文件编辑页面，如图 16-5 所示。



图 16-5　文件编辑界面

文件编辑是由 edit.php 实现，如代码 16-4 所示。

【代码 16-4】　edit.php

```
    <?php
      //省略部分代码
    if(!$allowbrowse) {
      print "  <tr><td class=error colspan=2>browsing under the explorer itself
is forbidden (see your administrator)</td></tr>\n";
      print "</table>\n</body></html>\n";
      exit();
    }
    ?>
    <tr>
      <td class=bdyForm>
       文件名:  <?php print $filename; ?>
      </td>
    </tr>
    <tr>
      <td class=bdyForm>
       <form method=POST action="save.php">
       <?php echo "<input type=hidden name=filename value='$filename'>\n"; ?>
       <textarea name=content cols=70 rows=20 style="font-family:arial; font-
size:9pt; color=#000000; border:1 solid #000000; background-color=#EFEF00"><?php
    if($content != "") echo urldecode($content);
```

```
    else {
        if(file_exists($filename)) {  //判断文件是否存在
            $fd=fopen($filename,"rb");  //以读模式打开
            $cts=fread($fd,filesize($filename));  //读入数据
            fclose($fd);
            echo $cts;  //输出文件内容
        }
    }
    ?></textarea>
    //……省略部分代码
```

该文件以读模式打开目标文本文件，并将文件内容显示在一个“textarea”中。

16.2.4 保存文件

在图 16-5 所示中，编辑完文件后，单击“Save”按钮，可以保存文件。保存文件的功能是由 save.php 来实现的，如代码 16-5 所示。单击“Save”按钮后，界面如图 16-6 所示。

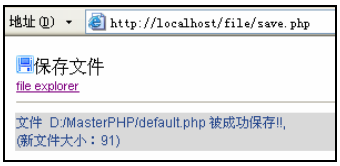


图 16-6 文件保存成功

【代码 16-5】 save.php

```
<?php
//省略部分代码
if($filename=="")
       Header("Location: unreadf.php?filename=$filename");
if(file_exists($filename) && is_writeable($filename) && !$IsSim) {
    /*获取编辑后的文件内容*/
    $content=$_POST['content'];
    /*打开文件，以写方式打开*/
    $fp=fopen($filename,"wb");
    /*write it*/
    $rc=fwrite($fp,$content,strlen($content));
    /*设定其访问模式*/
    chmod($filename,0644);
    /*close it*/
    fclose($fp);
}
?>
```

将编辑好的内容重新写入原来的文件，打开文件的时候要以“wb”方式打开。

16.2.5 通过浏览器浏览文件

文件管理系统除了能编辑文件外，还能够通过浏览器浏览一些常见的文本文件以及图像文件。在主界面中的文件名后的第 3 个按钮“”，表示可以通过浏览器来浏览文件。在图 16-1 所示中单击“4.jpg”后面的浏览按钮，则会直接打开浏览器，如图 16-7 所示。



图 16-7　浏览文件

浏览文件是由 send.php 来实现的，如代码 16-6 所示。

【代码 16-6】　send.php

```
<?php
//直接将内容发送到浏览器
header("Content-disposition: filename=".str_replace("/",'_',$filename).".src");
header("Content-type: application/octet-stream");  //内容类型
header("Pragma: no-cache");  //没有缓存
header("Expires: 0");  //不在客户保存
/*读取文件并发送到浏览器*/
readfile($ROOTPATH.$filename);
?>
```

16.2.6　创建目录

创建目录的功能在另一个浏览窗口中提供。用户在主界面中点击“更多功能”链接，便会进入辅助功能界面，如图 16-8 所示。



图 16-8　其他功能

显示更多功能页面的文件名为“lib/moreopt.php”，其代码可参见光盘。

在图 16-8 中的创建目录路径中输入“temp”，然后单击“mkdir”按钮，则会在指定路径下创建“temp”目录。比如本例在“d:\masterphp\file\”下创建 temp 目录（“d:\masterphp\file\”是服务器根目录）。

创建目录的功能由文件 mkdir.php 实现，如代码 16-7 所示。

【代码 16-7】　mkdir.php

```
<?php
//省略部分代码
if($dir=="")
      Header("Location: unreadf.php?dir=$dir");
if(is_dir($ROOTPATH.$dir)) {  //如果目录名已经存在，则提示错误
```

```
        $msg="failed: directory $dir already exists";
    }
    else {
        if($IsSim!=0) {
            $msg="simulation";
        }
        else {
            $msg="success";
            if(!mkdir($ROOTPATH.$dir,0755)) {  //调用 mkdir 创建目录
                $msg="failed";
            }
        }
    }
    //省略部分代码
```

16.2.7 上载文件

在图 16-8 中，在“上传文件到服务器”的“原文件”中，单击“浏览”按钮可以选择要上传的文件，在上传地址中输入地址，比如刚才创建的“/temp”目录，然后单击“Save”按钮即可上传。

上载文件的功能由 expl2save.php 实现，代码参见光盘。

16.2.8 删除目录

在图 16-8 所示的删除目录一栏中输入创建的“/temp”目录，然后单击“Close”按钮，可以将该目录删除。

删除目录的功能由文件 rmdir.php 实现，如代码 16-8 所示。

【代码 16-8】 rmdir.php

```
    <?php
    //省略部分代码
    /************************************************/
    if($IsSim!=0) { //是否处于模拟模式
        $msg="simulation";
    }
    else {
        $msg="success";
        if(!rmdir($ROOTPATH.$dir)) {  //调用 rmdir 删除目录
            $msg="failed";
        }
    }
    ?>
    //省略部分代码
```

16.2.9 设置权限

前面每个文件或者目录的处理页面中，都包含了一个 explorer.cfg 文件（如代码 16-9 所示），该文件包含了一些系统设置的权限。用户可以在这里设置权限，比如是否能够浏览目录，是否能够删除文件等。

【代码 16-9】　explorer.cfg

```
<?
/*模拟模式，表示系统处于演示，
如果要删除一个文件，则不会实际删除
(false=OFF,true=ON) ?*/
$IsSim=false;
/*是否显示删除链接?*/
$IsShowDelete=true;
/*是否显示文件详细信息 ?*/
$IsShowDetails=true;
/*是否允许浏览系统本身的文件*/
$AllowSelfBrowse=true;
/*文件保护，即不被显示的文件名
多个保护文件名之间用逗号分隔
如: $Exceptions = ".,..,passwd";
如果文件名未 passwd，则不会被显示，会被隐藏
*/
$Exceptions = ".,..,.htaccess,passwd";
?>
```

至此一个功能基本完善的文件管理系统介绍完毕，读者可以根据前面章节中介绍的文件处理函数，来扩充该系统的功能。

第 17 章　在线图片管理系统

本章将介绍一个功能比较完善的在线图片管理管理。通过该系统，用户能够很方便的实现图片库的管理。本系统具有如下一些功能：

（1）多用户管理；
（2）对图像的密码保护；
（3）文件上载到服务器；
（4）将图像数据保存进数据库；
（5）删除图片；
（6）编辑图片库；
（7）实现各种图片搜索；
（8）指定每页显示图片数；
（9）实现图像信息的快速浏览。

17.1　系统整体设计

根据对系统功能的分析，确定系统将由以下的文件组成：

（1）index.php，实现功能导航；
（2）include.php，用于数据库连接的设定以及常量的设定；
（3）show.php，用于显示图像、查询图像；
（4）showpic.php，用于从数据库中读取图像数据并发送到客户端浏览器；
（5）add.php，用于让用户选择图片，并将数据提交到服务器后保存到数据库中；
（6）delete.php，用于删除图片信息；
（7）edit.php，用于编辑图片信息；
（8）index.css，该文件为本系统的 CSS 样式文件；
（9）database.sql，该文件保存创建系统数据库的 SQL 语句。

17.2　创建数据库

系统数据库设计比较简单，只有一个数据库表格。首先要通过 MySQL 客户端创建一个名为“dbpic”的数据库，然后创建一个数据库表，表名为“picbase”，创建表格的 SQL 语句如下：

```
CREATE TABLE 'picbase' (
  'id' int(11) NOT NULL auto_increment,
  'who' text NOT NULL,
```

```
  'descr' text NOT NULL,
  'autor' text NOT NULL,
  'stamp' timestamp(14) NOT NULL,
  'pic' mediumblob NOT NULL,
  'thumb' blob NOT NULL,
  'pw' text NOT NULL,
  PRIMARY KEY  ('id'),
  UNIQUE KEY 'id' ('id')
);
```

读者可以使用前面章节介绍的方法创建数据库。

17.3　系 统 实 现

下面介绍整个系统的实现。

17.3.1　实现导航页面

导航页面由 index.php 实现（如代码 17-1 所示），该文件非常简单，主要是显示系统的功能列表。在浏览器中输入“http://localhost/pic/index.php”，自动进入到导航页面。显示结果如图 17-1 所示。

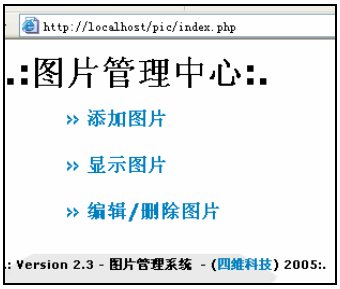


图 17-1　在线图片管理主界面

【代码 17-1】　index.php

```
<?php
include("include.php");
?>
<html>
<head>
    <title><?php echo $titleheader;?></title>
    <link rel="stylesheet" type="text/css" href="index.css">
</head>
<body>
<h1 align="center">.:图片管理中心:.</h1>
<h3><a href="add.php">&raquo; 添加图片</a></h3>
<h3><a href="show.php">&raquo; 显示图片</a></h3>
<h3><a href="show.php">&raquo; 编辑/删除图片</a></h3>
<?php echo $footer; ?>
</body>
</html>
```

17.3.2 设置系统参数

include.php 文件中声明了数据库的连接语句以及一些常量，也声明了一些常用的函数，如代码 17-2 所示。

【代码 17-2】 include.php

```
    <?php
    //变量定义
    $thumbsperline = "4";   //每行显示的缩略图的数量，0 表示 1 个，1 表示 2 个，依此类推
    $thumbsperpage = "5";      //每页显示的图像数
    $quality = "100";           //图像压缩比
    $titleheader = "在线图片管理系统";   //页面标题
    //页面尾部显示
    $footer = '<p class="copyright">.: Version 2.3 - 图片管理系统 - (<a href="forway
@zj.com">四维科技</a>) 2005:.</p>';
    //数据库参数设计
      $db_server = "localhost";
      $db_login = "root";
      $db_password = "123456";
      $db_name = "dbpic";
      $db_table = "picbase";
    //创建获取图像的缩略图
    function createthumb($filename,$thumbqual) {
        $size=getimagesize($filename);
        $width=$size[0];
        $height=$size[1];
        $newwidth=100;
        $newheight=intval($height*$newwidth/$width);
        $oldpic=ImageCreateFromJPEG($filename);
        $newpic=ImageCreate($newwidth,$newheight);
        ImageCopyResized($newpic,$oldpic,0,0,0,0,$newwidth,$newheight,$width,
$height);
        $newpicdata =  ImageJPEG($newpic,"temp.jpg",$thumbqual); //保存成临时文件
        $newpicfile = "temp.jpg";
        return $newpicfile;
    }
    //连接数据库函数
    function connect_database() {
        global $db_server, $db_login, $db_password;
        $db_connection=mysql_connect($db_server, $db_login, $db_password);
        if(! $db_connection){
            die("Error when connecting the database. Please check your login
information!");
        }
        return $db_connection;
    }
    //将十六进制转换成二进制
    function hex2bin($data) {
        $len = strlen($data);
        return pack("H" . $len, $data);
```

```
    }
    //将日期以普通字符输出
    function datumwandler($t) {
        return sprintf("%02d.%02d.%04d/%02d:%02d:%02d", substr($t, 6, 2), substr($t,
4, 2),substr($t, 0, 4), substr($t,8,2), substr($t,10,2), substr($t,12,2));
    }
    //替换重要字符，为了数据库的安全
    function prepare($value,$tagsallow) {
        $value = stripcslashes($value);
        $value = str_replace(chr(34),"&quot;",$value);     //替换双引号 "
        $value = str_replace(chr(39),"&rsquo;",$value);        //替换单引号 '
        if($tagsallow == "yes") {
        $value = strip_tags($value);
        }else{
            $value  = strip_tags($value,"<a><b><i><u><br><p><img>");
        }
        return $value;
    }
    //读取文件数据
    function gethex($file) {
        $fp = fopen ($file, "rb");      //打开图像文件
        $data = fread ($fp, filesize ($file)); //读取数据
        fclose($fp);                            //关闭
        $datahex = bin2hex($data);              //转换成二进制
        return $datahex;
    }
    ?>
```

该文件设置了数据库连接的一些参数，同时声明了常用的函数，这些函数能够方便的在其他功能模块中被调用。

17.3.3　添加图片

在主界面中单击“添加图片”链接，即可进入添加图片界面，如图 17-2 所示。



图 17-2　添加图片界面

用户单击“提交”按钮后，可以将图片以及相关数据保存到数据库中。选择文件可以通过单击“浏览”按钮来实现。本系统支持 JPG 文件，如果需要增加对 GIF 的支持，则需要单独下载 GIF 的支持包。“用户”一栏中，可以根据需要输入不同的用户名，这样方便管理和

查询。如果不设置密码，则图片是对外公开的，每个浏览者都可以查看和编辑。如果是比较重要的图片数据，最好加上密码。

提示 新版本的 PHP 因为 GIF 的版权问题，不再默认支持 GIF 图片的处理。

添加成功后，界面如图 17-3 所示。

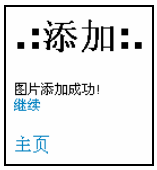


图 17-3 图片添加成功

添加图片的功能由文件 add.php 实现，如代码 17-3 所示。

【代码 17-3】 add.php

```
<?php
//包含文件
include("include.php");
?>
<html>
<head>
    <title><?php echo $titleheader;?> - Add a pic to databse</title>
    <script language="JavaScript" type="text/javascript">
    function check() {
        //检查密码是否一致
        if(document.add.pw.value != document.add.checkpw.value) {
            alert("Please enter the same password in both fields");
            return false;
        }else{
            return true;
        }
    }
    function picpreview() {
        //打开一个新窗口预览图片
        var pic = document.add.filename.value;
        window.open("add.php?action=picpreview&pic=" + pic,"","fullscreen");
    }
    </script>
    <link rel="stylesheet" type="text/css" href="index.css">
</head>
<body>
<h1>.:添加:.</h1>
<?php
$query = getenv("QUERY_STRING");
parse_url($query);
//用户单击预览链接的处理
if($action == "picpreview") {
    $pic = stripcslashes($pic);
```

```
        $info = getimagesize($pic);
        echo "<table cellpadding='10'>
        <tr>
            <td align='left'><b>Preview</b><br><br><a href='#' onclick= 'window.
close()'>Close window</a></td>
            <td align='right'><img src='$pic' $info[3]></td>
        </tr>
        </table>";
    }
    //如果第一次进入该页面，没有提交数据，则只显示 HTML 表单
    if($action == "") {
        //省略 html 表单界面
    }
    //如果用户单击提交按钮
    if($action == "add") {
        $db = connect_database();       //链接数据库
        $datahex = gethex($filename);  //获取 file 控件提交的数据
        $newpicfile = createthumb($filename,$quality);// 创建图片的缩略图
        $newpicdatahex = gethex($newpicfile); //将缩略图转换成二进制
        unlink($newpicfile);  //删除临时创建的文件
        $who = prepare($who,"no");          //获取提交的数据，通过 prepare 函数去掉一
些 html 标记
        $descr = prepare($descr,"yes");
        $author = prepare($author,"no");
        //创建插入图片数据的 sql 语句
        $cmd = "INSERT INTO '$db_table' ('id', 'who', 'descr', 'autor', 'stamp',
'pic', 'thumb', 'pw') VALUES ('', '$who ', '$descr', '$autor', NOW(NULL),
'$datahex', '$newpicdatahex', '$pw')";
        $result = mysql_db_query($db_name,$cmd); //query to the database
        // 执行成功
        if($result) {
            echo "<p>图片添加成功!<br><a href='index.php'>继续</a></p>";
        }else{      //if not
        echo "<p>Error:<br>";
        echo mysql_error($db);
    }
    mysql_close($db); //关闭数据库连接
    }
    ?>
    <a href="index.php">主页</a>
    </body>
    </html>
```

该文件主要用于显示添加图像的表单。当用户添加新图片之后，将图片的相关信息写入到数据库中，直接调用 insert 语句插入图片。



注意　获取图片文件数据是通过 gethex 函数实现的，该函数在 include.php 文件中声明。

17.3.4　查看图片

添加图片之后，就可以查看图片了。在主界面中，单击“显示图片”链接即可进入图片

查看界面，如图 17-4 所示。

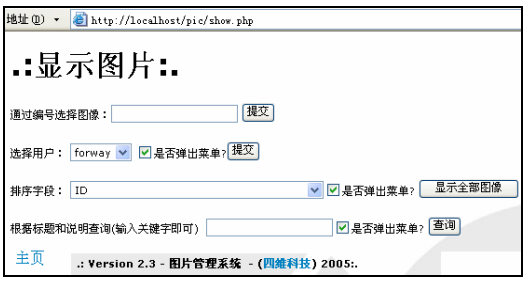


图 17-4　查看图片

从图 17-4 中可以看出，开始时用户并不能查看到图片，必须选择一定的查询方式才行。本系统可以通过多种方法来查看图像，可以根据图像的编号来查询图像、根据用户名来查看、查看所有图像、以及根据图像的说明或者标题来进行搜索。查询的时候支持关键字查询。

提示　复选框“是否显示弹出菜单”表示当用户在查看图片缩略图的时候，如果鼠标移动到缩略图时，是否自动弹出对话框显示图片的详细信息。

选择用户“forway”后，单击“提交”按钮，可以选中用户 forway 所有的图片，如图 17-5 所示。

因为在提交之前选中了“是否弹出菜单”复选框，因此将鼠标移至图片时，就会自动弹出该图片的相关信息，如图 17-6 所示。



图 17-5　图片显示结果

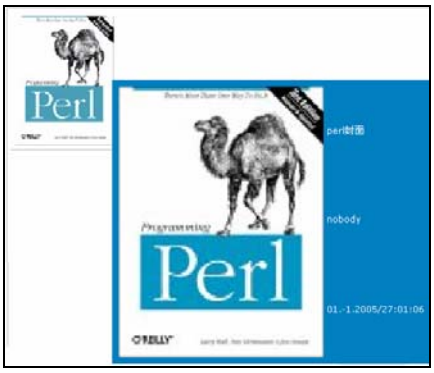


图 17-6　自动弹出图片相关信息

如果在提交图片的时候，设置了密码，则不能直接查看到该图片，而是用一个提示图片代替，如图 17-5 所示的最后一幅图片。图片中的文字为“Need Password”，表示需要密码。

单击任何一个没有加密的图片，可以查看该图片的详细信息，如图 17-7 所示。

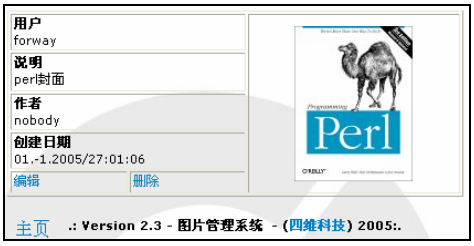


图 17-7　查看图片的详细信息

如果单击了加密的图片，则会提示输入密码，如图 17-8 所示。

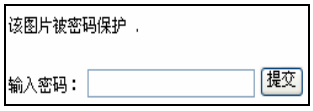


图 17-8　输入密码

输入密码后即可查看图片的详细信息。查看图片的功能由文件 show.php 实现，如代码 17-4 所示。

【代码 17-4】　show.php

```
    <?php
    //包含文件
    include("include.php");
    ?>
    <html>
    <head>
        <title><?php echo $titleheader;?> - Show Pics</title>
        <link rel="stylesheet" type="text/css" href="index.css">
    </head>
    <body>
    <?php
    $query = getenv("QUERY_STRING");  //获取提交方式
    parse_url($query);
    if($action == "") { //获取提交参数 action，action 为空时是默认方式，即第一次进入该页面
        $db = connect_database();  //连接数据库
        $cmd = "SELECT * FROM $db_table ORDER BY id;"; //sql 语句
        $erg = mysql_db_query($db_name,$cmd,$db);      //执行结果
        echo '<h1>.:显示图片:.</h1>';
        //输出让用户输入图片 id 的表单控件
        echo '<form action="show.php?action=byid" method="post" name="byid"
id="byid"><p>通过编号选择图像：<input type="text" name="id"> <input
type="submit" name="submit" value="提交"></p></form>';
        $array[0] = "<>"; //创建一个数组
        while (list($id,$who) = mysql_fetch_row($erg)) {   //从数据库中读取数据
                //如果数组中没有，则添加
                if(!in_array($who,$array)) {array_unshift($array,$who);    }
           }
        array_pop($array);  //去掉数组中没用的数据，即<>
        sort($array);       //排序
        //查看不同用户的图片
        echo '<form action="show.php?action=bywho&start=0" method="post"
name="bywho" id="bywho"><p>选择用户： <select name="who">';
        //循环输出所有用户
        for($x=0;$x<sizeof($array);$x++) {
            echo "<option value='$array[$x]'>$array[$x]</option>";
        }
        echo '</select> <input type="checkbox" name="popups" value="popups"
checked>是否弹出菜单?</input><input type="submit" name="submit" value="提交"></form>';
```

```
        //输出字段列表，让用户选择排序字段
        echo '<form action="show.php?action=showall&who=all&start=0" method=
"post" name="all" id="all">
        <p>排序字段： <select name="orderby">
            <option value="id" SELECTED>ID</option>
            <option value="who">Who</option>
            <option value="descr">Description (its stupid, becaus thats
text;-)</option>
            <option value="author">Author/Copyright</option>
            <option value="stamp">Timestamp</option>
        </select><input type="checkbox" name="popups" value="popups" checked>
是否弹出菜单?</input>
        <input type="submit" name="submit" value="显示全部图像"></p></form>';
        echo '<form action="show.php?action=search&start=0" method="post"
name="search" id="search">
        <p>根据标题和说明查询(输入关键字即可)  <input type="text"
name="who"><input type="checkbox" name="popups" value="popups" checked>是否弹出
菜单?</input> <input type="submit" name="submit" value="查询"></p>
        </form>';
        mysql_close($db);       //关闭数据库连接
    }
    //如果用户单击了查询按钮
    if($action == "search") {
        if($start == "*") {  //如果$start 为*，则表示从第 1 幅图片显示
            $cmd = "SELECT * FROM $db_table WHERE who LIKE '%$who%' OR descr LIKE
'%$who%' ORDER BY id;"; //如果是*号，则表示查询选定用户的所有图片
        }else{  //否则从第$start 幅图片开始显示
            $cmd = "SELECT * FROM $db_table WHERE who LIKE '%$who%' OR descr LIKE
'%$who%' ORDER BY id LIMIT $start,$thumbsperpage;";    //
        }
        $db=connect_database();     //连接数据库
        $cmdcount = "SELECT * FROM $db_table WHERE who LIKE '%$who%' OR descr LIKE
'%$who%';";
        $count = mysql_db_query($db_name,$cmdcount,$db);
        $numberofpics = mysql_num_rows($count);
        $message = "<h1>.:Search of $who:.</h1>";
        $message .="<h3>.:$who 总共有 $numberofpics 幅图片:.</h3>";
        if($start == "*") {
            $message .= "<h4>.:显示所有图片:.</h4>";
        }else{
            $aaa = $start + $thumbsperpage;
            $message .= "<h4>.:显示图片从： $start 到 $aaa:.</h4>";
        }
        $scrollquery = "search";
        showpics($db,$cmd,$db_name,$db_table,$thumbsperline,$thumbsperpage,$who,
$start,$message,$scrollquery,$orderby,$popups);
    }
    //如果是根据用户查询
    if($action == "bywho") {
        if($start == "*") { //判断起始显示图片编号
            $cmd = "SELECT * FROM $db_table WHERE who='$who' ORDER BY id;";
        //sql cmd to select all entries with the choosen person
```

```
        }else{
            $cmd = "SELECT * FROM $db_table WHERE who='$who' ORDER BY id LIMIT
$start,$thumbsperpage;";   //sql cmd to select all entries with the choosen perion
        }
        $db=connect_database();     //连接数据库
        $cmdcount = "SELECT * FROM $db_table WHERE who='$who';";
        $count = mysql_db_query($db_name,$cmdcount,$db);
        $numberofpics = mysql_num_rows($count);
        $message = "<h1>.:用户名称： $who:.</h1>";
        $message .=".:总共 $numberofpics 幅图片 :.";
        if($start == "*") {
            $message .= ".:显示所有图片:.";
        }else{
            $aaa = $start + $thumbsperpage;
            $message .= ".:图片显示从：$start 到 $aaa:.";
        }
        $scrollquery = "bywho";
        showpics($db,$cmd,$db_name,$db_table,$thumbsperline,$thumbsperpage,$who,
$start,$message,$scrollquery,$orderby,$popups);
    }
    //如果是显示所有图片
    if($action == "showall") {
        if($start == "*") {
            $cmd = "SELECT * FROM $db_table ORDER BY $orderby;";  //sql cmd to
select all entries with the choosen perion
        }else{
            $cmd = "SELECT * FROM $db_table ORDER BY $orderby LIMIT $start,
$thumbsperpage;";  //sql cmd to select all entries with the choosen perion
        }
        $db=connect_database();    //connect database
        $cmdcount = "SELECT * FROM $db_table;";
        $count = mysql_db_query($db_name,$cmdcount,$db);
        $numberofpics = mysql_num_rows($count);
        $message = "<h1>.:Show all pics:.</h1>";
        $message .=".:总图片数： $numberofpics:.";
        if($start == "*") {
            $message .= ".:显示所有图片:.";
        }else{
            $aaa = $start + $thumbsperpage;
            $message .= ".:图片显示从： $start 到 $aaa:.";
        }
        $scrollquery = "showall";

        showpics($db,$cmd,$db_name,$db_table,$thumbsperline,$thumbsperpage,
$who,$start,$message,$scrollquery,$orderby,$popups);
    }
    //显示图片的函数
    function
showpics($db,$cmd,$db_name,$db_table,$thumbsperline,$thumbsperpage,$who,$star
t,$message,$scrollquery,$orderby,$popups) {
        $who = trim($who);
        $whoscroll = $who;
```

```
        $a = "'";               //定义单引号，用于 javascript 中
        $erg = mysql_db_query($db_name,$cmd,$db);  //数据库查询
        //定义 javascript
        $datavor = 'if (num == |x|) num = text += |><tr><td><table
class="popup"><tr><td rowspan="3" width=20%><img src="showpic.php?id=|id|"
width="250"></td><td width="80%">|descr|</td></tr><tr><td
width="80%">|autor|</td></tr><tr><td
width="80%">|timestamp|</td></tr></table>|;';
        //用于显示快捷查看图片信息的动态 div 区域
        echo '
        <div id="describe" style="position:absolute; visibility:visible;
left:0px; top:-30px; z-index:2">
        </div>
        ';
        echo $message;
        echo '<table border="1" cellpadding="5" cellspacing="5" style="border-
collapse: collapse" bordercolor="#c0c0c0" width="100%">';
        $x=0;
        $y=0;
        while (list($id,$who,$descr,$autor,$timestamp,$pic,$thumb,$pw) = mysql_
fetch_row($erg)) {     //获得数据库数据
        if($y == "0") { echo "<tr>\n"; }
            if($pw == "") {     //判断是否需要密码
                if($popups) {
                    $timestamp = datumwandler($timestamp);
                    //定义时间输出格式
                    $descr = str_replace(chr(10),"<br>",$descr);
                    //将描述中的换行符号变成 html 中的<br>
                    $descr = str_replace(chr(13),"<br>",$descr);
                    $descr = str_replace("<br><br>","<br>",$descr);
                    $data[$x] = str_replace("|x|",$x,$datavor);
                        //用标准字符串显示
                    $data[$x] = str_replace("|id|",$id,$data[$x]);
                    $data[$x] = str_replace("|autor|",$autor,$data[$x]);
                    $data[$x] = str_replace("|descr|",$descr,$data[$x]);
                    $data[$x] = str_replace("|timestamp|",$timestamp,$data[$x]);
                    $data[$x] = str_replace("|","'",$data[$x]);
                    //output a link to the picture
                    echo "<td><a href='show.php?action=byid&id=$id'
onMouseover='popupLayer($x)' onMouseout='hideLayer(-30)'><img
src='showpic.php?id=$id&action=thumb' width='120' border='0'></a></td>\n";
                }else{
                    echo "<td><a href='show.php?action=byid&id=$id'><img
src='showpic.php?id=$id&action=thumb' width='120' border='0'></a></td>\n";
                }
            }else{      //如果需要密码
                echo "<td><a href='show.php?action=byid&id=$id'><img
src='pw.jpg' width='120' border='0'></a></td>\n";
            }
            if($y < $thumbsperline) { //判断是否需要换行显示图片，因为每行显示图片数可
以自己设定
                $y++;
```

```
            }else{
                echo "</tr>";
                $y = 0;
            }
            $x++;
        }
        echo "</table>";
        //数据 javascript
        if($popups) {  //如果允许快捷查看图片信息，则输出以下 javascript 语句
        echo '
        <script type="text/javascript"><!--
        var ns6 = (document.styleSheets&&!document.all)?1:0;
        var ie = document.all?1:0;
        function MouseHandler(e) {  //定义鼠标处理方式
            if (ns6) {  //如果是 netscape 浏览器
                Event = e;
                x = Event.pageX;
                y = Event.pageY;
            }
            if (ie) {  //ie 浏览器
                x = window.event.clientX;
                y = window.event.clientY;
            }
        }
        document.onmousemove = MouseHandler;
    //……省略部分 javascript 输出代码
    if($action == "byid") {     //根据图像编号查询
    ……//代码同前面的查询类似，省略
    }
    //如果有密码保护，输出"输入密码"的表单
    if($data[7] != "" AND $pw == "") {
        echo "<p>该图片被密码保护 .</p>";
        echo '<form action="show.php?action=byid&id=',$id,'" method="post"
name="pwform" id="pwform">
        <p>输入密码:  <input type="password" name="pw"> <input type=
"submit" name="submit" value="提交"></p>
        </form>';
    }
    //提交密码后，判断密码是否正确
    if($data[7] != "" AND $pw != "") {
        if($pw != $data[7]) {  //密码错误，则重新输入密码
            die("<p><a href='show.php?action=byid&id=$id'>Wrong password. Try
again</a></p>");
        }else{
            //否则显示图片，省略部分代码
        }
    }
    }
    ?>
    <?php echo $footer; ?>
    <a href="index.php">主页</a>
    </body>
```

```
    </html>
```

17.3.5 编辑图片

编辑图片和添加图片的界面一样，只是编辑图片界面中可以修改原来图片的一些信息，而不会增加一条记录。单击“编辑”链接，可以进入图像编辑界面，如图 17-9 所示。



图 17-9 图像编辑界面

用户可以在界面中编辑各种信息，如修改密码，重新设置密码，更换图片等。完成编辑后，单击“保存”按钮。

图片编辑由文件 edit.php 完成，如代码 17-5 所示。

【代码 17-5】 edit.php

```
    //由于大部分代码和添加图片类似，这里只给出一部分保存数据到数据库中的代码
    if($action == "save") {  //如果单击了保存按钮
        $db = connect_database();
        if($newpic == "no") {  //如果没有更换新图片
            $who = prepare($who,"no");              //去掉要写 html 标记
            $descr = prepare($descr,"yes");
            $author = prepare($author,"no");
            //更新数据
            $cmd = "UPDATE $db_table SET who='$who',descr='$descr',autor='$
author',stamp=NOW(NULL),pw='$pw' WHERE id=$id;";
            $result = mysql_db_query($db_name,$cmd);
            if($result) {
                echo "<p>Data updated.<br><a href='show.php?action=byid&id=$id'>
Go on</a></p>";
            }else{
                echo "<p>Error:</p>" . mysql_error();
            }
        }else{  //更换新图片
            $datahex = gethex($filename);  //获取新图片数据
```

```
            $newpicfile = createthumb($filename,$quality);// 创建缩略图
            $thumbhex = gethex($newpicfile); //获得缩略图数据
            unlink($newpicfile);  //删除临时文件
            $who = prepare($who,"no");
            $descr = prepare($descr,"yes");
            $author = prepare($author,"no");
            //更新数据
            $cmd = "UPDATE $db_table SET who='$who',descr='$descr', autor='$author',
stamp=NOW(NULL),pw='$pw',pic='$datahex',thumb='$thumbhex' WHERE id=$id;";
            $result = mysql_db_query($db_name,$cmd);
            if($result) {
                echo "<p>Data updated.<br><a href='show.php?action= byid&id=$id'>
Go on</a></p>";
            }else{
                echo "<p>Error:</p>" . mysql_error();
            }
        }
        mysql_close($db);
    }
```

17.3.6　删除图片

相对而言，删除图片最简单了，直接从数据库中删除对应的数据即可。单击“删除”链接，即可进入删除界面，如图 17-10 所示。

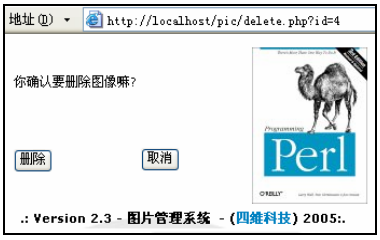


图 17-10　删除图片

删除图片的功能由文件 delete.php 完成，如代码 17-6 所示。

【代码 17-6】　delete.php

```
    <?php
    //省略部分代码
    if($action == "now") {  //如果 action 为 now，表示确认删除
        $db = connect_database();
        $cmd = "SELECT * FROM picbase WHERE id=$id;";
        $erg = mysql_db_query("dbpic",$cmd,$db);
        $data = mysql_fetch_row($erg);
        //判断是否有密码保护，如果有，则提示密码错误，重新输入
        if($data[7] != "" AND $pwsend == "") {
            echo "<p><a href='delete.php?id=$id'>Wrong password. Try again</a></p>";
        }else{
        if($data[7] == $pwsend) {  //密码正确
            //删除图片
            $cmd = "DELETE FROM $db_table WHERE id=$id;";
```

```
            $result = mysql_db_query($db_name,$cmd);
            if($result) {
                echo "<p>Pic number $id deleted sucessfully<br><a href='index.php'>
Continue</a></p>";
            }else{
                echo "<p>Error when deleting pic number $id" . mysql_error() .
"<br><a href='index.php'>Continue</a></p>";
            }
        }
        }
        mysql_close($db);
    }
    ?>
```

图片管理系统的核心功能就介绍到这里。读者可以在此基础之上做出更加完善的在线图片管理系统。

第 18 章　功能完善的 BBS 系统

BBS 是一个完整的网站系统不可缺少的模块。通过 BBS，用户能够发表自己的看法，可以回复别人的文章，从而进行知识共享，感情交流等。

本章将介绍一个功能完善的 BBS 论坛系统，该 BBS 系统具有以下功能：

（1）用户注册；
（2）用户登录；
（3）发表文章；
（4）回复文章；
（5）查看文章；
（6）统计文章；
（7）编辑文章；
（8）删除文章；
（9）取回密码；
（10）查看个人信息；
（11）热点文章显示；
（12）回复文章数量统计；
（13）快速功能导航；
（14）客户 IP 地址显示。

如果是以管理员账号登录，则可以实现更多的功能：

（1）创建讨论区；
（2）删除讨论区；
（3）编辑讨论区；
（4）删除用户；
（5）删除文章（可以删除所有文章）；
（6）权限授权。

下面介绍整个系统的实现。

18.1　系统整体设计

本系统功能比较强大，包含的文件也比较多，主要由以下文件组成。

（1）index.php，为论坛主页，用于显示讨论区的主要功能导航。
（2）config.php，为系统设置文件，用于数据库连接设置以及变量定义。
（3）install.php，为系统数据库安装文件，同时进行系统初始化设置。
（4）login.php，用于用户登录。

（5）register.php，用于系统注册。
（6）view_profile.php，用于查看个人信息设置。
（7）edit_profile.php，用于编辑个人信息。
（8）post_thread.php，用于发表新文章。
（9）post_reply.php，用于回复文章。
（10）view_message.php，用于查看文章。
（11）view_replies.php，用于查看回复文章。
（12）view_forums.php，用于查看论坛列表。
（13）view_threads.php，用于查看所有文章列表。
以上为普通用户文件，下面为管理员文件。
（1）forum_admin.php，用于论坛管理。
（2）general_admin.php，用于论坛通用属性设置。
（3）scheme_admin.php，用于论坛显示风格设置。
（4）user_admin.php，用于用户管理。
以上是本系统比较核心的文件，其他辅助文件，读者可以参考本书的光盘源代码。

18.2 数据库设计

BBS 系统由 8 个数据表组成，后台数据库采用 MySQL 5.0，数据库名称为“b2”。为了方便读者学习，数据库表是由系统自动安装完成的。

18.2.1 数据库表设计

（1）forums 表，该表保存论坛的基本信息，如论坛编号、论坛名称、论坛说明等。

```
create table forums
(
   forum_id int(10) not null auto_increment,  //论坛编号
   forum_order int(10) null,  //论坛序号
   forum_name varchar(64) not null,  //论坛名称
   forum_desc varchar(255) not null,  //论坛说明
   primary key(forum_id)
);
```

（2）threads 表，该表保存发表文章的基本信息，如文章编号、文章标题、发表时间、发表用户、用户 IP、论坛编号等。

```
create table threads
 (
   thread_id int(10) not null auto_increment,  //文章编号
   thread_title varchar(64) not null,  //文章题目
   thread_body text not null,  //文章内容
   thread_time timestamp(14) not null,  //发表时间
   user_id int(10) not null,  //发表用户
   user_ip varchar(15) not null,  //客户 IP 地址
   forum_id int(10) not null,  //论坛编号
   primary key(thread_id)
);
```

（3）users 表，该表保存注册用户的信息，包括用户名、密码、职业、主页、Email 等个人基本信息。

```
create table users
(
   user_id int(10) not null auto_increment,  //用户编号
   user_name varchar(64) not null,  //用户名
   user_email varchar(128) not null,  //Email
   user_pass varchar(64) not null,  //密码
   user_location varchar(128) null,  //地址
   user_occupation varchar(64) null,  //职业
   user_homepage varchar(128) null,  //主页
   user_picture varchar(128) null,  //头像
   user_interests varchar(255) null,  //爱好
   user_aim varchar(16) null,  //msn
   user_icq varchar(16) null,  //oicq
   user_yahoo varchar(32) null,  //yahoo 账号
   user_signature varchar(255) null,  //用户签名
   user_usesig int(1) null default '0',  //是否使用签名
   primary key(user_id)
);
```

（4）replies 表，该表保存回复文章的信息，包括回复文章编号、文章内容、回复时间、被回复的原始文章编号等信息。

```
create table replies
(
   reply_id int(10) not null auto_increment,  //回复编号
   reply_body text not null,  //回复内容
   reply_time timestamp(14) not null,  //回复时间
   user_id int(10) not null,  //发表用户
   user_ip varchar(15) not null,  //用户 IP 地址
   thread_id int(10) not null,   //原始文章编号
   forum_id int(10) not null,  //论坛编号
   primary key(reply_id)
);
```

（5）moderators 表，该表用于版主管理，版主由管理员授权。

```
create table moderators
(
   moderator_id int(10) not null auto_increment,  //版主编号
   user_id int(10) not null,  //用户编号
   forum_id int(10) not null,  //论坛编号
   primary key(moderator_id)
);
```

（6）administrators 表，该表保存管理员信息，如管理员编号、用户编号。管理员详细信息保存在 user 表中。

```
create table administrators
(
   admin_id int(10) not null auto_increment,  //管理员编号
   user_id int(10) not null,   //用户编号
   primary key(admin_id)
);
```

（7）messages 表，该表用于保存会员之间的信息发送

```
create table messages
(
   message_id int(10) not null auto_increment,  //信息编号
   message_name varchar(64) not null,  //信息名称
   message_body text not null,  //信息内容
   primary key(message_id)
);
```

（8）schemes 表，该表保存系统的一些属性设置、如背景颜色、文字颜色和表格颜色等。

```
create table schemes
(
   scheme_id int(10) not null auto_increment,
   scheme_name varchar(64) not null,
   scheme_desc varchar(255) not null default 'No description provided.',
   background_color varchar(7) not null default '#FFFFFF',
   table_border_color varchar(7) not null default '#000000',
   table_border_size int(1) not null default '1',
   header_background varchar(7) not null default '#FFFFFF',
   menu_background varchar(7) not null default '#EEEEEE',
   text_color varchar(7) not null default '#000000',
   text_font varchar(64) not null default 'Verdana',
   text_small int(2) not null default '10',
   text_regular int(2) not null default '12',
   link_color varchar(7) not null default '#000000',
   table_header_background varchar(7) not null default '#000000',
   table_header_text_color varchar(7) not null default '#FFFFFF',
   table_color_1 varchar(7) not null default '#EEEEEE',
   table_color_2 varchar(7) not null default '#CCCCCC',
   error_message varchar(7) not null default '#FF0000',
   active_scheme varchar(1) not null default '0',
   primary key(scheme_id)
);
```

（9）properties 表，该表保存整个 BBS 系统的信息，如 BBS 的名称、BBS 系统的标志（LOGO）等。

```
create table properties
(
   board_name varchar(64) not null default 'Forums',  //论坛名称
   title_image varchar(128) not null default './images/title.png' //论坛标志
);
```

18.2.2 数据库安装配置

为了方便读者学习，本书设计了自动安装程序，用户只需要输入相关的属性就可以方便地完成整个数据库的创建。

首先需要启动 MySQL 客户端，然后创建数据库“b2”，如图 18-1 所示。

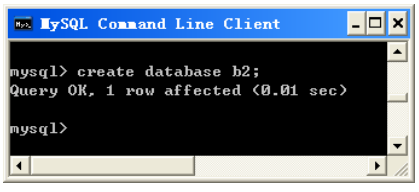


图 18-1 创建数据库 b2

接着运行数据库安装程序。要将光盘中对应的“BBS”目录拷贝到系统根目录下，然后在浏览器中输入“http://localhost/bbs/include/index.php”，会显示系统安装

界面。出于安全性考虑，连接到该页面后，系统会自动跳转到“http://localhost/bbs/index.php”页面，但是如果直接连接“http://localhost/bbs/index.php”，则系统会提示错误。连接成功后，显示界面如图 18-2 所示。



图 18-2　系统安装界面

在图 18-2 中，可以进行多种自定义的设置。在“MySQL 属性”区域可以输入连接数据库的基本信息。笔者设置的用户名是“root”、密码是“123456”、数据库是前面创建的“b2”，表前缀表示在创建的时候，所有的表名前面都加上“b2_”，用户可以自定义设计。

另外，在安装的时候需要创建管理员的用户名和密码，这里输入的管理员账号是“admin”，密码是“123456”。

最开始的时候，是没有论坛的，因此安装时需要初始化一个论坛。此处创建了一个“汽车版”论坛。一切输入完毕后，单击“开始安装”，进入如图 18-3 所示界面。

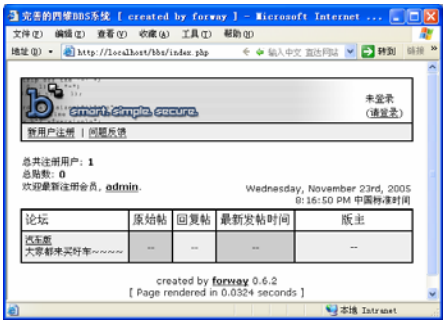


图 18-3　系统安装完成界面

经过以上步骤后，系统已经安装完毕，可以使用该 BBS 论坛了。安装完成后，系统会自动在当前目录下创建一个名为“config.php”的文件（如代码 18-1 所示），该文件保存了创建时的一些设置信息。

【代码 18-1】 config.php

```
<?
/* 定义常量 INSTALLED 为 yes，表示系统已经安装 */
define("INSTALLED", "yes");
if (INSTALLED == "yes")
{
    /* MySQL 数据库参数 */
    define("DB_USER", "root");
    define("DB_PASS", "123456");
    define("DB_NAME", "b2");
    define("DB_HOST", "localhost");
    /* 数据库表的前缀*/
    define("TABLE_PREFIX", "b2_");
    /*是否提示错误信息 */
    define("ADMIN_ERRORS", "no");
    /* 提示管理员是否有错误 */
    define("NOTIFY_ADMIN", "yes");
    /* 管理员 Email */
    define("ADMIN_EMAIL", "admin@sohu.com");

}
?>
```

如果进入 MySQL 客户端，可以查看到系统已经创建了 8 个数据库表，如图 18-4 所示。

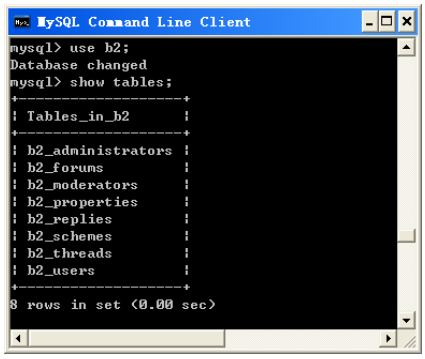


图 18-4 查看数据库 b2 包含的表

提示 创建数据库的文件为“install.php”，该文件在“index”中被包含。

创建数据库表的语句如下：

```
/* Check to see if config.php is present */
if ( !@include("config.php") )
{
    /* No config? then call the installer! */
    require("install.php");
```

```
    exit;
}
/* If config.php is there, then check the installation status */
else if (INSTALLED != "yes")
{
    /* Not installed? then call the installer! */
    require("install.php");
    exit;
}
```

以上代码检查是否目录中已经有“config.php”文件，如果没有则调用 install.php 文件创建数据库。同时分析全局变量 INSTALLED 的值是否为“yes”，如果不是“yes”也要重新安装数据库。



如果用户不小心删除了数据库“b2”想重新安装数据库，则需要进行两方面设置。第一方面是直接删除 config.php 文件，然后执行最开始的安装步骤。第二方面是在 config.php 文件中将 INSTALLED 的值设定为“no”，然后执行最开始的安装步骤。

下面简单介绍用于创建数据库的文件 install.php，该文件自动创建数据库的所有表格，如代码 18-2 所示。

【代码 18-2】　install.php

```
<?php
/* 包含初始化数据文件 */
require("./language/installer.php");
/* 设定系统安装文件名 */
$file_name = "install.php";
/* 获得文件名的长度 */
$file_name_length = -(strlen($file_name));
/* 检查文件名是否正确，这样保证安全，防止用于非法安装 */
if (substr($_SERVER['SCRIPT_NAME'], $file_name_length) == $file_name)
  header("Location: ../index.php");
/* Turn off dummy error messages */
define("ADMIN_ERRORS", "yes");
/* 设定安装步骤 */
$step = "";
/* 获得数据的提交方法 */
$REQUEST_METHOD = $_SERVER['REQUEST_METHOD'];
/* 如果是用户提交，即 POST 方法提交 ... */
if ( $REQUEST_METHOD == "POST" )
  {
    /* 定义数据库的参数 */
    define("DB_USER", $_POST['db_user']);
    define("DB_PASS", $_POST['db_pass']);
    define("DB_NAME", $_POST['db_name']);
    define("DB_HOST", $_POST['db_host']);
    $table_prefix = $_POST['table_prefix'];
    /* 获得步骤 */
    $step = $_POST['step'];
    /* 系统安装分成 3 步，如果不是第 3 步，获取提交信息 */
```

```
      if ($step != 3)
        {
          $board_name  = $_POST['board_name'];
          $title_image = $_POST['title_image'];
          $admin_user  = $_POST['admin_user'];
          $admin_pass  = $_POST['admin_pass'];
          $admin_pass2 = $_POST['admin_pass2'];
          $admin_email = $_POST['admin_email'];
          $forum_name = $_POST['forum_name'];
          $forum_desc = $_POST['forum_desc'];
        }
      /* 创建 config.php 文件*/
      $config_file = "<?
   /*定义安装状态为 yes，再次执行该文件，则不会重新安装*/
   define(\"INSTALLED\", \"yes\");
   if (INSTALLED == \"yes\")
     {
       //……省略部分代码，该代码创建 config.php 的文件内容
     }

   ?>";
     }
   /* 根据不同的步骤显示不同的内容 */
   switch ($step)
     {
      /* 第 1 步，显示初始化表单*/
      default:
      case 1:
        /* 显示页面头部 html 内容 */
        ShowHeader();
        //省略创建 html 表单内容，表单界面见图 18-2
        /* 显示页面尾部内容 */
        ShowFooter();
        break;
      /* 第 2 步，创建数据库 */
      case 2:
        /* Show the HTML header */
        ShowHeader();
        /* 连接数据库 */
        define("CONNECTION", mysql_connect(DB_HOST, DB_USER, DB_PASS));
        /* 如果不成功 */
        if (!CONNECTION)
          exit("<CENTER>数据库连接有问题.<BR><BR><A href=\"javascript: history.go
(-1)\">返回</CENTER>\n");
        else
          {  //如果连接成功，则创建数据库表
            /* 创建'forums' 表 */
            $SQL = "create table " . $table_prefix . "forums
                    (
                      forum_id int(10) not null auto_increment,
                      forum_order int(10) null,
                      forum_name varchar(64) not null,
```

```
                    forum_desc varchar(255) not null,
                    primary key(forum_id)
                  );";
            //省略其他创建表的代码
             $results = ExeSQL($SQL);  //执行
            //插入初始化数据，初始化论坛版块，如前面创建的"汽车版"
             $SQL    = "INSERT INTO " . $table_prefix . "forums (forum_order,
forum_name, forum_desc) VALUES ('0', '$forum_name', '$forum_desc');";
             $results = ExeSQL($SQL);
             /* 设定论坛标志，默认为 title.png */
             if ($title_image == "")
               $title_image = "./images/title.png";
            //……省略其他初始化 sql 语句
             ShowFooter();
          }
        mysql_close(CONNECTION);
        break;
      /*第 3 步，显示 config.php 文件 */
      case 3:
        header("Content-Type: text/x-delimtext; name=\"config.php\"");
        header("Content-disposition: attachment; filename=config.php");
        echo $config_file;
        exit();
        break;
    }
  function  ShowHeader()  //页面头
  {
      //代码省略
  }
  function ShowFooter()  //页面尾
  {
      //代码省略
  }
  ?>
```

18.3　系 统 实 现

由于整个系统实现的功能比较多，在此只会介绍一些核心的代码，其余部分，请参阅本书的光盘。

18.3.1　系统注册/登录

在图 18-3 所示中，如果用户不登录，是不能发表文章的，但是可以查看文章。在图 18-3 中单击“请登录”的链接，会进入到登录界面。如果用户没有注册，需要在界面中单击“新用户注册”链接，进入注册界面。

在注册之前，会显示会员注册的一些提示信息和注意事项等，如图 18-5 所示。

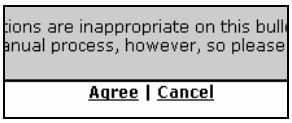


图 18-5　注册注意事项

在图 18-5 中，单击“Agree”链接表示同意，然后进入注册界面，如图 18-6 所示。

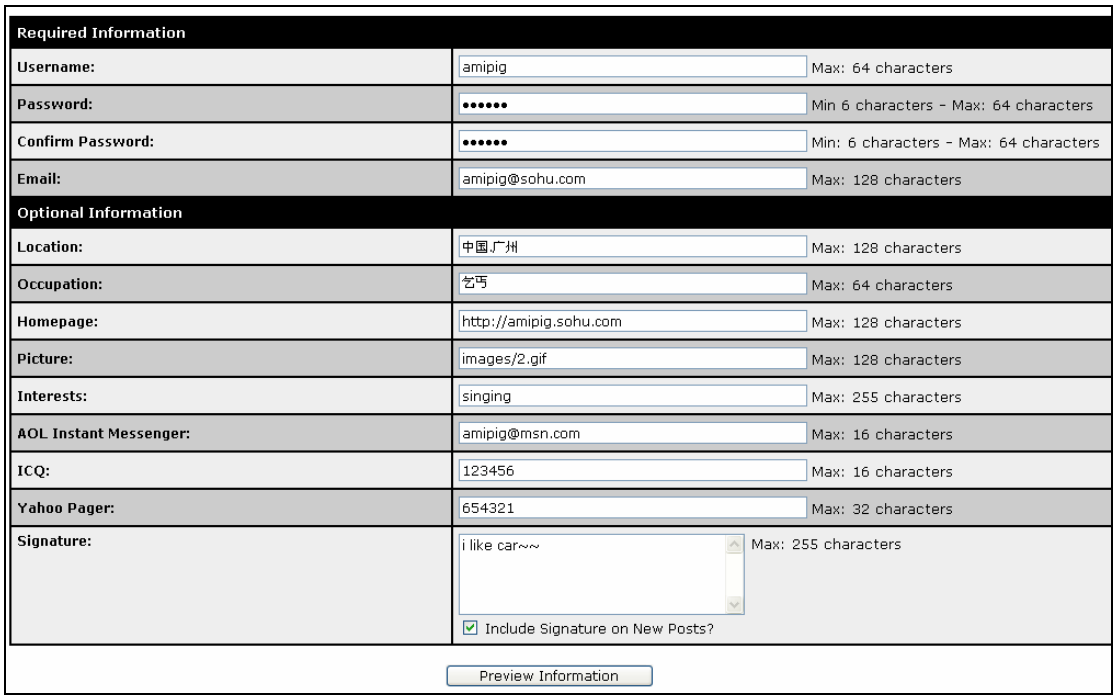


图 18-6　新用户注册界面

单击“Preview Information”按钮，可以预览注册信息，如图 18-7 所示。

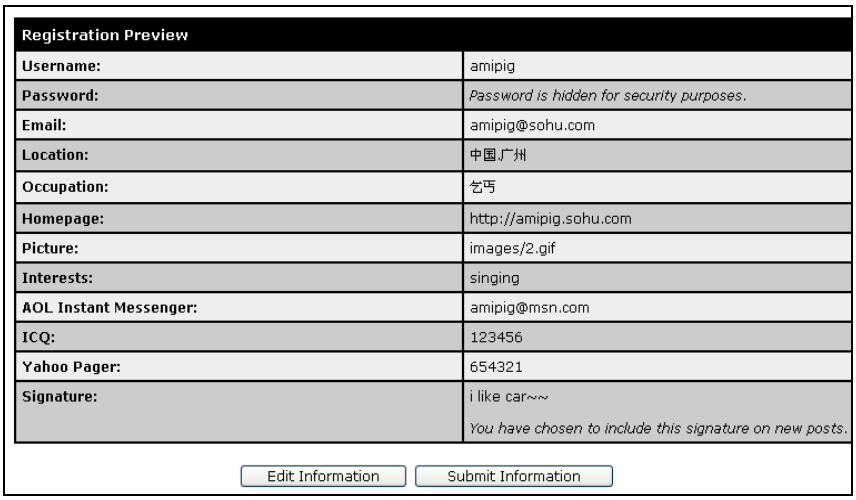


图 18-7　注册信息预览

如果发现有输入错误，可以单击“Edit Information”按钮重新编辑。如果确认无误，则可以单击“Submit Information”按钮提交到数据库中。注册成功后，系统会提示欢迎新注册会员，如图 18-8 所示。

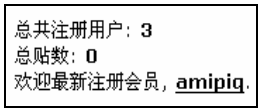


图 18-8　会员注册成功

会员注册主要由文件“content/register.php”来实现，由于该文件代码比较长，此处只给出数据库操作的代码以及重要的函数定义，如代码 18-3 所示。

【代码 18-3】　register.php

```
    //……省略部分代码
    /* 接下来按照不同步骤显示不同内容 */
    switch ($step)
      {
           default:
        case 1:  //第 1 步，显示系统协议，注意实事项等信息
         //省略部分代码
          /* 完成，输出 agree 和 ACancel 按钮 */
          echo "              </TD>\n"
             . "            </TR>\n"
             . "          </TABLE>\n"
             . "          <CENTER class=\"regular_text\">\n"
             . "            <B><A href=\"?pid=register&step=2\">Agree</A> | <A
href=\"?pid=view_forums\">Cancel</A></B>\n"
             . "          </CENTER>\n";
          break;
        /* 第 2 步，显示注册表单*/
        case 2: //调用 ShowRegistrationForm 显示注册表单
          ShowRegistrationForm( $username, $password, $confirm_password, $email,
$location, $occupation, $homepage, $picture, $interests, $aim, $icq, $yahoo,
$signature, $include_sig );
          break;
        case 3:  //第 3 步，预览注册信息
            //省略部分代码
          /* 显示必填信息 */
          PreviewSection ( $username, "Username", &$the_color );
          PreviewSection ( $password, "Password", &$the_color );
          PreviewSection ( $email, "Email", &$the_color );
          /* 显示选填信息 */
          if ( $location != "" )
            PreviewSection( $location, "Location", &$the_color );
         //省略其他字段的预览信息
          break;
        /* 第 4 步，提交到数据库 */
        case 4:
          /* 检测提交方法，如果是通过 POST 方法提交，说明用户单击了提交按钮 */
          if ( $REQUEST_METHOD == "POST" )
            {
              /* No errors... yet */
              $no_err = 0;
              /* 查询该用户名是否存在*/
              $SQL = "SELECT COUNT(*) as user_exists FROM " . TABLE_PREFIX . "users
WHERE user_name='$username';";
              $results = ExeSQL($SQL);
              /* Grab the data, parse the results */
              while ($row = mysql_fetch_array($results))
                {
```

```
                /* 如果用户名存在 */
                if ($row["user_exists"] != 0)
                  {
                    echo "          <CENTER class=\"error_message\">That
username already exists!</CENTER><BR>\n";
                    $no_err++;
                  }
              }
             /* 判断电子邮件是否已经存在 */
             $SQL    = "SELECT COUNT(*) as email_exists FROM " . TABLE_PREFIX .
"users WHERE user_email='$email';";
             $results = ExeSQL($SQL);
             /* Grab the data, parse the results */
             while ($row = mysql_fetch_array($results))
               {
                 /* 如果存在则提示错误 */
                 if ($row["email_exists"] != 0)
                   {
                     echo "          <CENTER class=\"error_message\">Someone has
already registered using that email address!</CENTER><BR>\n";
                     $no_err++;
                   }
               }
             /* 如果没有错误*/
             if ($no_err == 0)
               {
                 /* Clear out the URL variables if they still contain 'http://' */
                 if ($homepage == "http://") { $homepage = ""; }
                 if ($picture == "http://") { $picture = ""; }
                 /* 对密码进行加密 */
                 $password = crypt($password);
                 /*是否自动添加签名 */
                 if ($include_sig != 1)
                   $include_sig == 0;
                 /* 插入到数据库 */
                 $SQL    = "INSERT INTO " . TABLE_PREFIX . "users (user_name,
user_email, user_pass, user_location, user_occupation, user_homepage,
user_picture, user_interests, user_aim, user_icq, user_yahoo, user_signature,
user_usesig) VALUES ('$username', '$email', '$password', '$location',
'$occupation', '$homepage', '$picture', '$interests', '$aim', '$icq', '$yahoo',
'$signature', '$include_sig');";
                 $results = ExeSQL($SQL);
                 /* 设置用户的 cookie*/
                 SetCookie("user_name", $username, time() + 86400, '',
$_SERVER['HTTP_HOST']);
                 SetCookie("user_pass", $password, time() + 86400, '', $_SERVER
['HTTP_HOST']);
                 /* 设置登录状态*/
                 $logged_in = 1;
                 /* 完成注册 */
                 echo "          <CENTER class=\"regular_text\">\n"
                    . "           <B>Thanks for registering!</B><BR>\n"
```

```
                    . "            <A href=\"index.php\">Click here to log
in!</A>\n"
                    . "          </CENTER>\n"
                    . "          <BR>\n";
                 require("./content/view_forums.php");
                 return;
               }
             else
    //…..省略部分代码
```

注册系统首先判断用户名和电子邮件是否存在，如果存在则提示重新注册，然后将密码进行加密，最后将数据保存进数据库中。

提示　注册成功后，需要登录系统，只有登录后，才能发表文章以及进行其他的操作。另外管理员和普通会员登录后，主界面显示的菜单是不一样的。

在主界面上，单击“请登录”链接，将会出现会员登录界面，如图 18-9 所示。

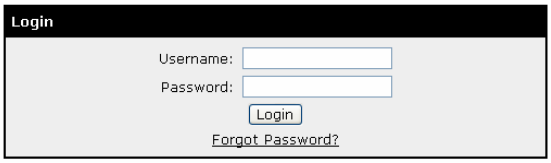


图 18-9　会员登录界面

在表单中输入用户名和密码，然后单击“Login”按钮，可以登录。如果忘记密码，可以单击“Forgot Password?”链接找回密码。

首先以普通会员“amipig”的身份注册登录，显示的功能菜单如图 18-10 所示。



图 18-10　普通会员登录后的界面

如果以管理员账号“admin”登录，则会显示更多的功能菜单。在图 18-10 中单击“Log Out”注销登录，然后以管理员身份进行登录，登录成功后的界面如图 18-11 所示。



图 18-11　管理员登录后的界面

提示　显示哪一个菜单是在系统的主文件“index.php”中根据登录权限进行设置的。

系统登录界面由文件“content/login.php”来实现，如代码 18-4 所示。

【代码 18-4】　login.php

```
<?
//……省略部分代码
```

```
/*如果登录失败，显示登录失败信息 */
if ($login == "failed")
 echo "         <CENTER class=\"error_message\">登录失败，请重新登录</CENTER><BR>";
/* 显示登录对话框*/
echo "           <FORM method=\"POST\" action=\"index.php?pid=login\">\n"
// ……省略部分显示登录表单显示代码
?>
```

这里要注意的是，login.php 主要用于显示登录对话框，而登录是否正确，由“index.php?pid=login”来进行处理。index.php 文件是一个综合处理文件，将在后面介绍。

18.3.2 发表文章

既然是 BBS 系统，则发表文章是最重要的功能。任何一个注册会员都能够发表文章。在主界面中单击任何一个论坛标题，即可进入该论坛内查看文章。例如在图 18-3 中单击“汽车版”链接，即可进入该论坛，如图 18-12 所示。



图 18-12 查看文章

因为系统刚刚创建，所以没有任何已发表的文章。在图 18-12 所示中单击“Click here 发表帖子”进入发表文章界面，如图 18-13 所示。

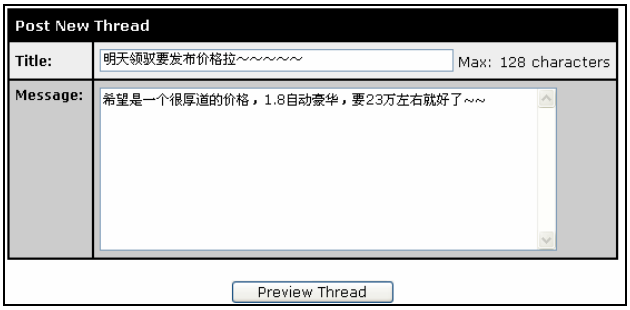


图 18-13 发表文章

输入内容后，可以单击“Preview Thread”进行预览，如图 18-14 所示。

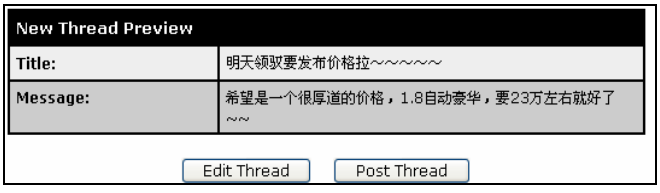


图 18-14 预览新文章

单击“Post Thread”按钮，可以显示文章列表，如图 18-15 所示。

发表文章，主要是通过文件“content /post_thread.php”来实现，如代码 18-5 所示。

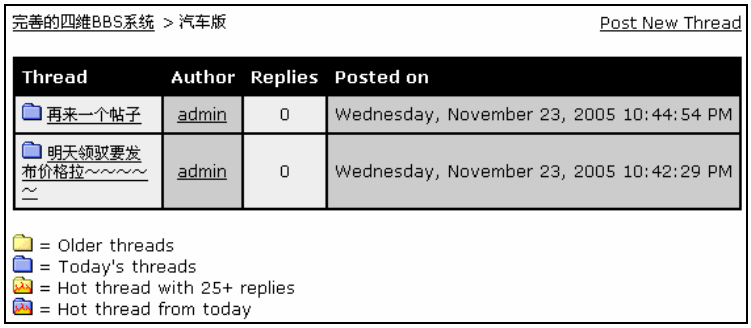


图 18-15　显示文章列表

【代码 18-5】　post_thread.php

```
    <?php
    //省略部分代码
    /*发表文章分成以下几个步骤 */
    switch ($step)
      {
        /* 第 1 步，显示发表文章的表单 */
        default:
        case 1:
          ShowPostThreadForm( $username, $password, $email, $title, $message,
$include_sig, $user_id, $forum_id );
          break;
        /* 显示发表文章的预览信息 */
        case 2:
          /*显示文章预览信息 */
          echo "          <FORM action=\"?pid=post_thread\" method=\"POST\" name=\
"post_thread\">\n"
             . "            <TABLE cellpadding=\"5\" cellspacing=\"0\"
width=\"100%\" border class=\"table_border\">\n"
             . "              <TR class=\"table_header\">\n"
             . "                <TD colspan=\"2\">New Thread Preview</TD>\n"
             . "              </TR>\n"
             . "              <TR bgcolor=\"" . TABLE_COLOR_1 . "\" class=\"regular_
text\">\n"
             . "                <TD width=\"25%\"><B>Title:</B></TD>\n"
             . "                <TD width=\"50%\">\n"
             . "                  $title\n"
             . "                  <INPUT type=\"hidden\" name=\"title\" value=\
"$title\">\n"
             . "                </TD>\n"
             . "              </TR>\n"
             . "              <TR bgcolor=\"" . TABLE_COLOR_2 . "\"
class=\"regular_text\">\n"
             . "                <TD width=\"25%\"
valign=\"top\"><B>Message:</B></TD>\n"
             . "                <TD width=\"50%\">\n";

          /* 添加用户的签名 */
```

```
        $SQL    = "SELECT user_signature FROM " . TABLE_PREFIX . "users WHERE
user_id='$user_id';";
        $results = ExeSQL($SQL);
        /* 获取签名 */
        while ($row = mysql_fetch_array($results))
          $signature = $row["user_signature"];
        /* 如果有，则添加*/
        if ($signature != "" && $include_sig == "yes")
          $display_message = $message . "<BR><BR>" . $signature;
        else
          $display_message = $message;
        /* 显示提交按钮，包括 Edit Thread 和 Post Thread */
        echo "                 $display_message\n"
           . "                 <INPUT type=\"hidden\" name=\"message\"
value=\"$message\">\n"
           . "               </TD>\n"
           . "             </TR>\n"
           . "           </TABLE>\n"
           . "           <INPUT type=\"hidden\" name=\"include_sig\"
value=\"$include_sig\">\n"
           . "           <INPUT type=\"hidden\" name=\"forum_id\"
value=\"$forum_id\">\n"
           . "           <INPUT type=\"hidden\" name=\"user_id\"
value=\"$user_id\">\n"
           . "           <CENTER>\n"
           . "             <BR>\n"
           . "             <INPUT type=\"Submit\" value=\"Edit Thread\" name=\
"action\">\n"
           . "              \n"
           . "             <INPUT type=\"Submit\" value=\"Post Thread\" name=\
"action\">\n"
           . "           </CENTER>\n"
           . "         </FORM>\n";
        break;
       /* 检查用户输入，然后将数据保存进数据库*/
       case 3:
        /* 确保为 Post Thread 按钮提交的数据 */
        if ( $REQUEST_METHOD == "POST" )
          {
           /* 选择用户签名 */
           $SQL    = "SELECT user_signature FROM " . TABLE_PREFIX . "users WHERE
user_id='$user_id';";
           $results = ExeSQL($SQL);
           /* 获取签名 */
           while ($row = mysql_fetch_array($results))
             $signature = $row["user_signature"];
           /* 包含签名 */
           if ($signature != "" && $include_sig == "yes")
             $message = $message . "<BR><BR>" . $signature;
           /* 将文章插入数据库 */
           $SQL    = "INSERT INTO " . TABLE_PREFIX . "threads (thread_title,
thread_body, user_id, user_ip, forum_id) VALUES ('$title', '$message', '$user_id',
```

```
'$user_ip', '$forum_id');";
            $results = ExeSQL($SQL);
            /* 显示保存完成信息 */
            echo "          <CENTER class=\"normal_message\">Thanks for
posting!</CENTER><BR>\n";
            /* 显示文章列表，该功能由 view_threads.php 实现*/
            require ("./content/view_threads.php");
          }
         else
          {
            /*如果不是提交而来，则显示发表文章的表单*/
            echo "          <CENTER class=\"error_message\">Malformed request
detected!</CENTER><BR>\n";
            //显示发表文章的表单界面
             ShowPostThreadForm( $username, $password, $email, $title, $message,
$include_sig, $user_id, $forum_id );
          }
         break;
     }
    //省略部分代码
    ?>
```

发表文章时首先验证用户是否登录，然后显示发表文章的表单，用户输入完成后，可以进行预览。如果确认无误，提交后，系统会检查用户是否有签名，并且是否设置了“自动添加到文章中”，如果设置了自动添加，则会将签名添加到发表的文章中。查看文章列表由 view_threads.php 实现。

18.3.3　查看文章列表

在发布文章的时候，我们已经看到了查看文章列表，即当用户选择了一个讨论区之后，可以看到所有的文章。在图 18-15 所示中，“Older threads”表示的是旧帖子，图标是黄色的；“Today's threads”表示今天的帖子，图标是蓝色的；“Hot Thread with 25+ replies”表示回复文章超过 25 篇的热门帖子；而“Hot thread from today”表示的是当天的热门文章。这样区分容易让使用者捕捉到当前论坛的热点。

例如某汽车论坛的界面如图 18-16 所示。



图 18-16　某汽车论坛的讨论区

在文章列表中，可以清楚看到每个文章的标题、回复数、发布者，以及最后发布时间等信息。显示文章列表是由文件“content/ view_threads.php”实现的，如代码 18-6 所示。

【代码 18-6】　view_threads.php

```
<?
//省略部分代码
```

```
    /* 首先获取所有论坛版信息*/
    $SQL    = "SELECT forum_id FROM " . TABLE_PREFIX . "forums;";
    $results = ExeSQL($SQL);
    /* 将所有论坛版块编号保存到一个数组中 */
    while ($row = mysql_fetch_array($results))
      $forum_list[] = $row["forum_id"];
    /* 如果论坛不存在，则停止*/
    if (!in_array($forum_id, $forum_list))
      {
        /* Tell the user what's up */
        echo "         <CENTER class=\"error_message\">论坛不存在 !</CENTER><BR>\n";
        require ("./content/view_forums.php");
        return;
      }
    /* 开始显示表格 */
    echo "          <TABLE width=\"100%\" cellpadding=\"0\" cellspacing=\"0\"
>\n"
       . "            <TR>\n";
    /* 从论坛数据库中读取数据 */
    $SQL    = "SELECT * FROM " . TABLE_PREFIX . "forums WHERE forum_id='$forum_id';";
    $results = ExeSQL($SQL);
    /*读取并显示*/
    while ($row = mysql_fetch_array($results))
      echo "              <TD class=\"regular_text\"><A href=\"?pid=view_forums\">" .
BOARD_NAME . "</A> > " . $row["forum_name"]."</TD>\n";
    /* 获取论坛总文章数量 */
    $SQL    = "SELECT COUNT(*) AS any_threads FROM " . TABLE_PREFIX . "threads
WHERE forum_id='$forum_id';";
    $results = ExeSQL($SQL);
    /* 获取文章发表 */
    while ($row = mysql_fetch_array($results))
      $any_threads = $row["any_threads"];
    /* 显示发表文章链接，如果已经有文章*/
    if ($any_threads != 0)
      {
        /* 显示发表新文章链接 */
        echo "              <TD align=\"right\" class=\"regular_text\"><A href=\"?
pid=post_thread&forum_id=$forum_id\">Post New Thread</A></TD>\n"
          . "            </TR>\n"
          . "          </TABLE>\n"
          . "          <BR>\n";
        /* 显示表头 */
        echo "          <TABLE cellspacing=\"0\" cellpadding=\"5\" width=\"100%\"
border class=\"table_border\">\n"
          . "            <TR class=\"table_header\">\n"
          . "              <TD width=\"100%\">Thread</TD>\n"
          . "              <TD align=\"center\" width=\"1\">Author</TD>\n"
          . "              <TD align=\"center\" width=\"1\">Replies</TD>\n"
          . "              <TD width=\"\" nowrap>Posted on</TD>\n"
          . "            </TR>\n";
        /* 从论坛文章列表中获取每个文章的信息，按照时间进行排序*/
        $SQL    = "SELECT *, DATE_FORMAT(thread_time, '%W, %M %e, %Y %r') AS
```

```
nice_time, DATE_FORMAT(thread_time, '%Y-%m-%d') AS post_date FROM " . TABLE_PREFIX .
"threads WHERE forum_id='$forum_id' ORDER BY thread_time DESC;";
       $results = ExeSQL($SQL);
       /*获取数据并显示 */
       while ($row = mysql_fetch_array($results))
         {
          /*获取时间*/
          $current_date = strftime ("%Y-%m-%d", time());
          /* 获取文章编号和用户编号 */
          $thread_id = $row["thread_id"];
          $user_id   = $row["user_id"];
        /* 获取每篇文章的回复数量 */
        $SQL     = "SELECT COUNT(*) AS total_items FROM " . TABLE_PREFIX . "replies
WHERE thread_id='$thread_id';";
        $results2 = ExeSQL($SQL);
        /* 将数据保存到数组 */
        while ($row2 = mysql_fetch_array($results2))
         $total_items = $row2["total_items"];
          /* 显示回复文章数 */
          if ($total_items == "")
            $total_replies = "--";
          else
            $total_replies = $total_items;
        /* 根据用户编号来获取用户名称 */
        $SQL     = "SELECT * FROM " . TABLE_PREFIX . "users WHERE user_id='$user_
id';";
        $results2 = ExeSQL($SQL);
        /* 获取数据 */
        while ($row2 = mysql_fetch_array($results2))
         $the_user = $row2["user_name"];
          /* 根据文章的热门程度，显示不同的图标 */
          if ( $row["post_date"] == $current_date && $total_replies >= 25 )
            $which_image = "folder-blue-fire";
          else if ( $row["post_date"] == $current_date )
            $which_image = "folder-blue";
          else if ( $total_replies >= 25 )
            $which_image = "folder-yellow-fire";
          else
            $which_image = "folder-yellow";
          /* 显示具体信息 */
          echo "            <TR>\n"
             . "<TD bgcolor=\"" . TABLE_COLOR_1 . "\" class=\"regular_text\">\n"
             . "<IMG src=\"./images/$which_image.png\"> <A
href=\"?pid=view_replies&thread_id=" . $row["thread_id"] .
"&forum_id=$forum_id\">" . $row["thread_title"] . "</A>\n"
             . " </TD>\n"
             . "<TD bgcolor=\"" . TABLE_COLOR_2 . "\" align=\"center\" nowrap
class=\"small_text\">\n"
             . "<A href=\"?pid=view_profile&user=" . $the_user . "\">" . $the_user .
"</A>\n"
             . "</TD>\n"
             . "<TD bgcolor=\"" . TABLE_COLOR_1 . "\" align=\"center\" class=\
```

```
"regular_text\">\n"
             . "" . $total_replies . "\n"
             . "</TD>\n"
             . "<TD bgcolor=\"" . TABLE_COLOR_2 . "\" nowrap class=\"small_text\">\n"
             . "" . $row["nice_time"] . "\n"
             . "</TD>\n"
             . "</TR>\n";
        }
       /* 文章图标说明信息 */
       echo "</TABLE>\n"
          . "<BR>\n"
          . "<FONT class=\"small_text\">\n"
          . "<IMG src=\"./images/folder-yellow.png\"> = Older threads<BR>\n"
          . "<IMG src=\"./images/folder-blue.png\"> = Today's threads<BR>\n"
          . "<IMG src=\"./images/folder-yellow-fire.png\"> = Hot thread with 25
+ replies<BR>\n"
          . "<IMG src=\"./images/folder-blue-fire.png\"> = Hot thread from
today<BR>\n"
          . " </FONT>\n";
     }
    else
     {
       /*如果总数为 0，则显示本论坛暂时没有文章 */
       echo "</TR>\n"
          . " </TABLE>\n"
          . " <BR>\n"
          . " <CENTER class=\"regular_text\">\n"
          . "  <B>本论坛暂时没有帖子，欢迎灌水~~.</B><BR>\n"
          . "<A href=\"?pid=post_thread&forum_id=$forum_id\">Click here 发表帖
子.</A>\n"
          . "</CENTER>\n";
     }
    ?>
```

18.3.4 查看文章

进入文章列表后，虽然能够查看更多的消息，但是并不能查看每个文章的具体内容，要查看文章的详细信息，必须点击每个文章的标题，然后查看详细信息，如图 18-17 所示。

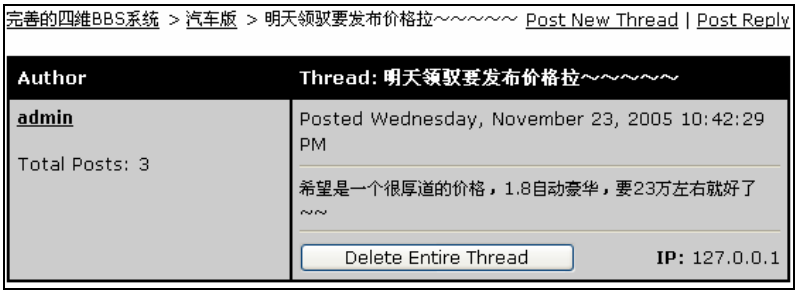


图 18-17 查看文章详情

从图 18-17 中，可以详细看到文章的标题、内容、文章的发布者，以及发布人的 IP 地址来源，同时还可以查看发布人的详细信息。

提示　在查看文章的时候，同时会有一个“Delete Entire Thread”按钮，该按钮用来删除整个帖子以及所有回复的文章。只有管理员才具备这个权限，而普通用户是看不到该按钮的。

查看文章的功能由文件“content/view_replies.php”实现，如代码 18-7 所示。

【代码 18-7】　view_replies.php

```
<?php
//省略部分代码
/*创建一个数组，用来保存论坛版主信息 */
$moderated_forums[] = "0";
/* 判断该用户是否为一个版主 */
$SQL    = "SELECT * FROM " . TABLE_PREFIX . "moderators WHERE user_id='$user_id';";
$results = ExeSQL($SQL);
/* 获取数据 */
while ($row = mysql_fetch_array($results))
  $moderated_forums[] = $row["forum_id"];
/* 获取论坛编号 */
$SQL    = "SELECT forum_id FROM " . TABLE_PREFIX . "forums;";
$results = ExeSQL($SQL);
/* 将论坛列表保存进数组 */
while ($row = mysql_fetch_array($results))
  $forum_list[] = $row["forum_id"];
/* 从文章列表数据库中查询出文章编号 */
$SQL    = "SELECT thread_id FROM " . TABLE_PREFIX . "threads;";
$results = ExeSQL($SQL);
/*保存进数据库 */
while ($row = mysql_fetch_array($results))
  $thread_list[] = $row["thread_id"];
/* 如果论坛不存在，则停止输出 */
if ( !in_array($forum_id, $forum_list) || !in_array($thread_id, $thread_list) )
  {
    echo "          <CENTER class=\"error_message\"><B>论坛不存在!</CENTER>
<BR>\n";
    require ("./content/view_forums.php");
    return;
  }
echo "         <TABLE width=\"100%\" cellpadding=\"0\" cellspacing=\"0\">\n"
   . "            <TR>\n";
/* 获取论坛名称*/
$SQL    = "SELECT * FROM " . TABLE_PREFIX . "forums WHERE forum_id='$forum_id';";
$results = ExeSQL($SQL);
/* 显示论坛名称 */
while ($row = mysql_fetch_array($results))
  echo "            <TD class=\"regular_text\"><A href=\"?pid=view_forums\">" .
BOARD_NAME . "</A> > <A href=\"?pid=view_threads&forum_id=" . $row["forum_id"] . "\">" .
$row["forum_name"] . "</A> > ";
/* 获取文章信息 */
$SQL    = "SELECT * FROM " . TABLE_PREFIX . "threads WHERE thread_id=$thread_id;";
$results = ExeSQL($SQL);
```

```
    /* 显示文章标题 */
    while ($row = mysql_fetch_array($results))
      echo "          " . $row["thread_title"] . "</TD>\n";
    /* 显示回复文章的链接，以及发表新文章的链接 */
    echo "          <TD align=\"right\" class=\"regular_text\"><A href=\"?pid=
post_thread&forum_id=$forum_id\">Post New Thread</A> | <A href=\"?pid=post_reply
&thread_id=$thread_id&forum_id=$forum_id\">Post Reply</A></TD>\n"
       . "        </TR>\n"
       . "       </TABLE>\n"
       . "       <BR>\n";
    /* 获取文章所有信息 */
    $SQL    = "SELECT * FROM " . TABLE_PREFIX . "threads WHERE
thread_id='$thread_id';";
    $results = ExeSQL($SQL);
    /* 保存到数组 */
    while ($row = mysql_fetch_array($results))
      $thread_topic = $row["thread_title"];
    /* 通过表格显示 */
    echo "          <TABLE cellspacing=\"0\" cellpadding=\"5\" width=\"100%\"
border class=\"table_border\">\n"
       . "            <TR class=\"table_header\">\n"
       . "              <TD width=\"150\">Author</TD>\n"
       . "              <TD width=\"100%\">Thread: $thread_topic</TD>\n"
       . "            </TR>\n";
    /* 获取要被显示的的文章信息 */
    $SQL    = "SELECT *, DATE_FORMAT(thread_time, '%W, %M %e, %Y %r') AS nice_time
FROM " . TABLE_PREFIX . "threads WHERE thread_id='$thread_id' ORDER BY
thread_title;";
    $results = ExeSQL($SQL);
    /* 从数据库中获取发表该文章的用户名称 */
    while ($row = mysql_fetch_array($results))
      {/* Pull each user name from the database */
        $SQL     = "SELECT * FROM " . TABLE_PREFIX . "users WHERE user_id='" .
$row["user_id"] . "';";
        $results2 = ExeSQL($SQL);
        while ($row2 = mysql_fetch_array($results2))
          {
            $user_name    = $row2["user_name"];
            $user_location = $row2["user_location"];
          }
        /*获取该用户总共发表文章数 */
        $SQL     = "SELECT COUNT(*) AS total_posts FROM " . TABLE_PREFIX . "threads
WHERE user_id='" . $row["user_id"] . "';";
        $results2 = ExeSQL($SQL);
        /*保存 */
        while ($row2 = mysql_fetch_array($results2))
          $total_posts = $row2["total_posts"];
        /* 获取该用户回复文章数量*/
        $SQL     = "SELECT COUNT(*) AS total_posts FROM " . TABLE_PREFIX . "replies
WHERE user_id='" . $row["user_id"] . "';";
        $results2 = ExeSQL($SQL);
        /* 获取发布文章的总数量 */
```

```
       while ($row2 = mysql_fetch_array($results2))
         $total_posts = $total_posts + $row2["total_posts"];
        //显示用户信息
       echo "            <TR>\n"
          . "              <TD bgcolor=\"" . TABLE_COLOR_2 . "\" width=\"200\"
valign=\"top\" nowrap>\n"
          . "              <FONT class=\"regular_text\"><B><A href=\"?pid=view_
profile&user=$user_name\">$user_name</A></B></FONT><BR><BR>\n"
          . "                <FONT class=\"small_text\">\n"
          . "                  Total Posts: $total_posts<BR>\n";
       /* 显示用户位置 */
       if ($user_location != "") { echo "                Location: $user_location
<BR>\n"; }
       /* 显示文章发布时间*/
       echo "                </FONT>\n"
          . "              </TD>\n"
          . "              <TD bgcolor=\"" . TABLE_COLOR_2 . "\" width=\"100%\"
valign=\"top\">\n"
          . "                <FONT class=\"small_text\">Posted " .
$row["nice_time"] . "</FONT>\n"
          . "                <HR>\n"
          . "                <FONT class=\"regular_text\">\n"
          . "                  " . $row["thread_body"] . "\n"
          . "                </FONT>\n";
       /* 如果该用户是版主或者管理员 */
       if ( ( $is_moderator != 0 && in_array($forum_id, $moderated_forums) ) ||
$is_admin != 0 )
         {
          /* 则显示每个发表文章或者回复文章用户的 IP 地址*/
          $SQL     = "SELECT user_ip FROM " . TABLE_PREFIX . "threads WHERE
thread_id=" . $row["thread_id"] . ";";
          $results2 = ExeSQL($SQL);
          while ($row2 = mysql_fetch_array($results2))
            $user_ip = $row2["user_ip"];
          echo "               <HR>\n"
             . "               <TABLE width=\"100%\" cellspacing=\"0\" cellpa-
dding=\"0\">\n"
             . "                 <TR class=\"small_text\">\n";
          /* 判断是版主还是管理员 */
          if ($is_admin == 0)
            $which = "mod";
          else
            $which = "admin";
          /* 显示表单 */
          echo "                   <FORM action=\"index.php\" method=\"POST\"
name=\"" . $which . "_tools\">\n"
             . "                     <TD>\n"
             . "                       <INPUT type=\"hidden\" name=\"forum_id\"
value=\"$forum_id\">\n";
          /* 检查是否有回复文章编号 */
          if (isset($row["reply_id"]))
            echo "                       <INPUT type=\"hidden\"
```

```
name=\"reply_id\" value=\"" . $row["reply_id"] . "\">\n";
          /*通过隐藏表单保存一些文章的相关信息' */
          echo "                      <INPUT type=\"hidden\" name=\"thread_id\"
value=\"" . $row["thread_id"] . "\">\n"
            . "                       <INPUT type=\"submit\" name=\"" . $which .
"_action\" value=\"Delete Entire Thread\" onClick=\"return Confirm('Are you sure
you want to delete this thread, and all of the associated replies?');\">\n"
            . "                     </TD>\n"
            . "                     <TD align=\"right\">\n"
            . "                       <B>IP:</B> " . $user_ip . "\n"
            . "                     </TD>\n"
            . "                   </FORM>\n"
            . "                 </TR>\n"
            . "               </TABLE>\n";
        }
       echo "              </TD>\n"
         . "            </TR>\n";
     }
    /* 获取每篇回复文章的信息 */
    $SQL    = "SELECT *, DATE_FORMAT(reply_time, '%W, %M %e, %Y %r') AS nice_time
FROM " . TABLE_PREFIX . "replies WHERE thread_id='$thread_id' ORDER BY
reply_time;";
    $results = ExeSQL($SQL);
    /* 获取数据 */
    while ($row = mysql_fetch_array($results))
     {
       /* 获取用户信息 */
       $SQL     = "SELECT * FROM " . TABLE_PREFIX . "users WHERE user_id='" .
$row["user_id"] . "';";
       $results2 = ExeSQL($SQL);
       /* 保存到变量中 */
       while ($row2 = mysql_fetch_array($results2))
         {
           $user_name    = $row2["user_name"];
           $user_location = $row2["user_location"];
         }
       /* 获取发表文章数*/
       $SQL     = "SELECT COUNT(*) AS total_posts FROM " . TABLE_PREFIX . "threads
WHERE user_id='" . $row["user_id"] . "';";
       $results2 = ExeSQL($SQL);
       while ($row2 = mysql_fetch_array($results2))
         $total_posts = $row2["total_posts"];
       /* 获取回复文章数*/
       $SQL     = "SELECT COUNT(*) AS total_posts FROM " . TABLE_PREFIX . "replies
WHERE user_id='" . $row["user_id"] . "';";
       $results2 = ExeSQL($SQL);
       /* 发表文章总数 */
       while ($row2 = mysql_fetch_array($results2))
         $total_posts = $total_posts + $row2["total_posts"];
       echo "           <TR>\n"
         . "             <TD bgcolor=\"" . TABLE_COLOR_1 . "\" width=\"150\"
valign=\"top\" nowrap>\n"
```

```
          . "               <FONT class=\"regular_text\"><B><A
href=\"?pid=view_profile&user=$user_name\">$user_name</A></B></FONT><BR><BR>\
n"
          . "               <FONT class=\"small_text\">\n"
          . "                 Total Posts: $total_posts<BR>\n";
       /* 显示用户地址 */
       if ($user_location != "") { echo "                Location:
$user_location<BR>\n"; }
       /* 显示回复时间和回复内容*/
       echo "               </FONT>\n"
          . "             </TD>\n"
          . "             <TD bgcolor=\"" . TABLE_COLOR_1 . "\"
width=\"100%\">\n"
          . "               <FONT class=\"small_text\">Posted " .
$row["nice_time"] . "</FONT>\n"
          . "               <HR>\n"
          . "               <FONT class=\"regular_text\">\n"
          . "                 " . $row["reply_body"] . "\n"
          . "               </FONT>\n";
       /* 如果是管理员或者版主，则显示更多功能按钮*/
       if ( ( $is_moderator != 0 && in_array($forum_id, $moderated_forums) ) ||
$is_admin != 0 )
        {
          /* 显示回复文章的用户的 IP 地址*/
          $SQL     = "SELECT user_ip FROM " . TABLE_PREFIX . "replies WHERE
reply_id='" . $row["reply_id"] . "';";
          $results2 = ExeSQL($SQL);
          /* 保存 */
          while ($row2 = mysql_fetch_array($results2))
            $user_ip = $row2["user_ip"];
          /* 显示 */
          echo "               <HR>\n"
             . "               <TABLE width=\"100%\" cellspacing=\"0\"
cellpadding=\"0\">\n"
             . "                 <TR class=\"small_text\">\n";
          /* 判断是管理员还是版主*/
          if ($is_admin == 0)
            $which = "mod";
          else
            $which = "admin";
          /*显示表单，如删除回复文章的按钮 */
          echo "                   <FORM action=\"index.php\" method=\"POST\"
name=\"" . $which . "_tools\">\n"
             . "                     <TD>\n"
             . "                       <INPUT type=\"hidden\" name=\"forum_id\"
value=\"$forum_id\">\n"
             . "                       <INPUT type=\"hidden\" name=\"thread_id\"
value=\"$thread_id\">\n"
             . "                       <INPUT type=\"hidden\" name=\"reply_id\"
value=\"" . $row["reply_id"] . "\">\n"
             . "                       <INPUT type=\"submit\" name=\"" . $which .
"_action\" value=\"Delete Reply\" onClick=\"return Confirm('Are you sure you want
```

```
to delete this reply?');\">\n"
             . "                    </TD>\n"
             . "                    <TD align=\"right\" valign=\"middle\">\n"
             . "                      <B>IP:</B> $user_ip\n"
             . "                    </TD>\n"
             . "                  </FORM>\n"
             . "                </TR>\n"
             . "              </TABLE>\n";
        }
       echo "             </TD>\n"
          . "           </TR>\n";
     }
    echo "          </TABLE>\n";
    ?>
```

18.3.5 回复文章

既然是讨论区，就应该能够让用户对其他用户发表的文章进行评论。点击进入任何一篇文章后，除了可以查看文章之外，还可以回复文章。

一般要先查看文章之后再回复，因此只有打开具体某个文章的时候，系统才会显示回复链接。单击如图 18-17 所示的“Post Reply”链接，即可进入回复界面，如图 18-18 所示。

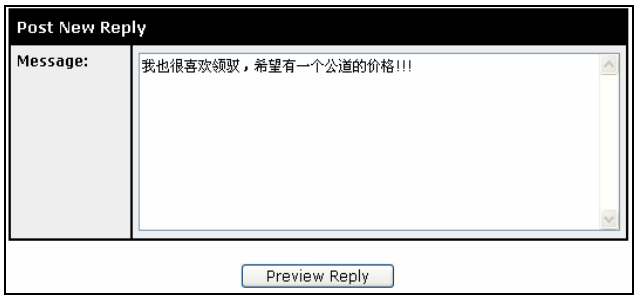


图 18-18 回复文章界面

输入回复信息后，可以单击“Preview Reply”按钮预览回复，如图 18-19 所示。

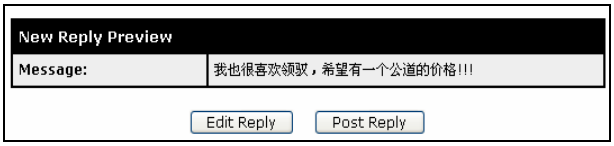


图 18-19 预览回复

如果确认无误后，可以单击“Post Reply”发布回复文章，回到原文章的界面，如图 18-20 所示。

回复文章的功能由文件“content/ post_reply.php”实现，如代码 18-8 所示。

【代码 18-8】 post_reply.php

```
    <?php
    /*回复文章和发表文章原理类似，这里省略其代码。需要注意的是，回复文章的时候需要向 replies
表中插入数据，插入数据的时候，要保存原始文章的编号。*/
    ?>
```



图 18-20　显示回复的文章

18.3.6　查看个人信息

当用户登录后，会显示“个人信息”的菜单，单击可以直接查看用户的基本信息设置，如图 18-21 所示。单击“Edit Profile”可以编辑个人信息。

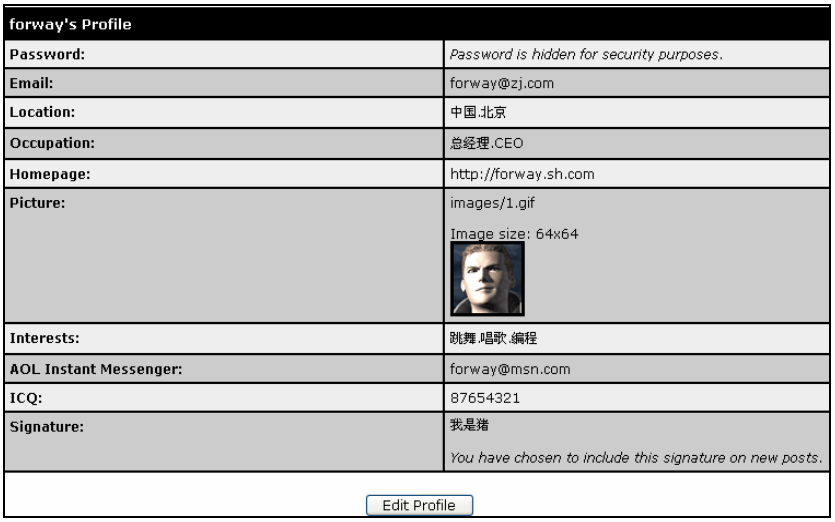


图 18-21　查看个人信息

在论坛中，单击任何一个发文章的用户名称，也可以查看该用户的简单信息。例如在图 18-20 中单击“admin”，可以查看用户“admin”简单信息，同图 18-21 所示类似，只是没有“Edit Profile”按钮。

查看个人信息是由 view_profile.php 实现的。而编辑个人信息是由 edit_profile.php 文件实现的，由于和会员注册类似，这里省略其代码。

18.3.7　实现管理员功能

由于参与 BBS 讨论的人无法核实身份，因此对于有些不当的言论，或者故意捣乱的会员，需要管理员予以禁止或者删除。

要实现管理员功能，首先要以管理员身份登录，管理员账号是在系统安装的时候设定，用户名“admin”，密码“123456”。

1．讨论区通用属性

在管理员菜单中，单击“通用管理”，能够设置 BBS 系统的名称以及设置 BBS 系统的 LOGO。设置完后，系统名称以及系统 LOGO 就会被应用到网站中。比如登录到网站后，最上面显示的图像就是系统的图像，而系统名称会在多处被使用，如快捷链接，如图 18-22 所示。其中“完善的四维 BBS 系统”就是系统名称。

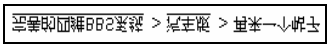


图 18-22　快捷链接

提示　快捷链接方便用户快速导航，尤其是进入到网站的多层链接之后，可以快速地回到想去的板块。

通用属性的设置界面如图 18-23 所示。

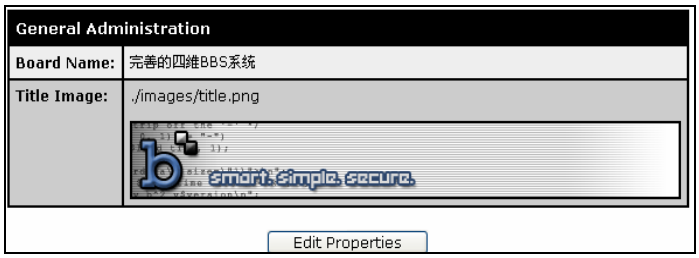


图 18-23　通用属性的设置

单击“Edit Properties”按钮，可以编辑该设置，如图 18-24 所示。

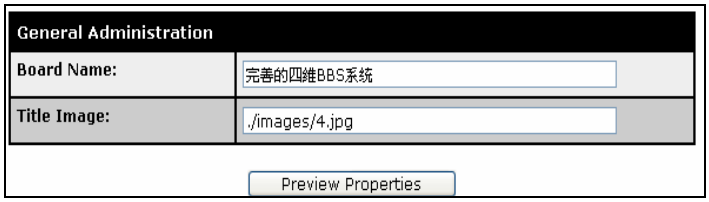


图 18-24　编辑通用属性

此处将论坛图像改成“4.jpg”，然后单击“Preview Properties”按钮，显示的结果如图 18-25 所示。



图 18-25　修改后的系统 LOGO

单击“Submit Properties”按钮，可以保存该修改。

进行通用设置的功能是由文件 general_admin.php 实现的，代码可参见光盘。

2．系统设置

为了让系统更加美观，许多方面都提供了自定义的功能，如整个讨论区的显示风格、表格显示、文字显示、图像显示等都可以进行自定义的设置。

在管理员菜单中，单击“系统设置”链接，可以进入系统设置界面，如图 18-26 所示。

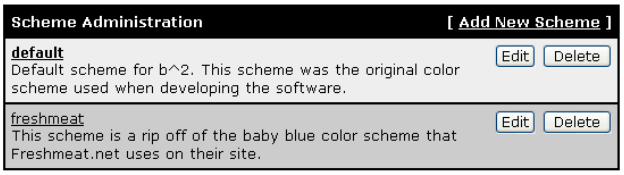


图 18-26　系统设置界面

本系统提供了两种论坛显示方案，管理员可以单击“default”或者“fleshmeat”来更换不同风格。另外还可以编辑或者删除这些系统设置。

用户可以新增显示风格，单击“Add New Scheme”进行相关添加。

在图 18-27 中，选中“Scheme is active?”复选框，可以设定为当前默认风格。单击“Preview Information”按钮可以预览属性设置，如图 18-28 所示。

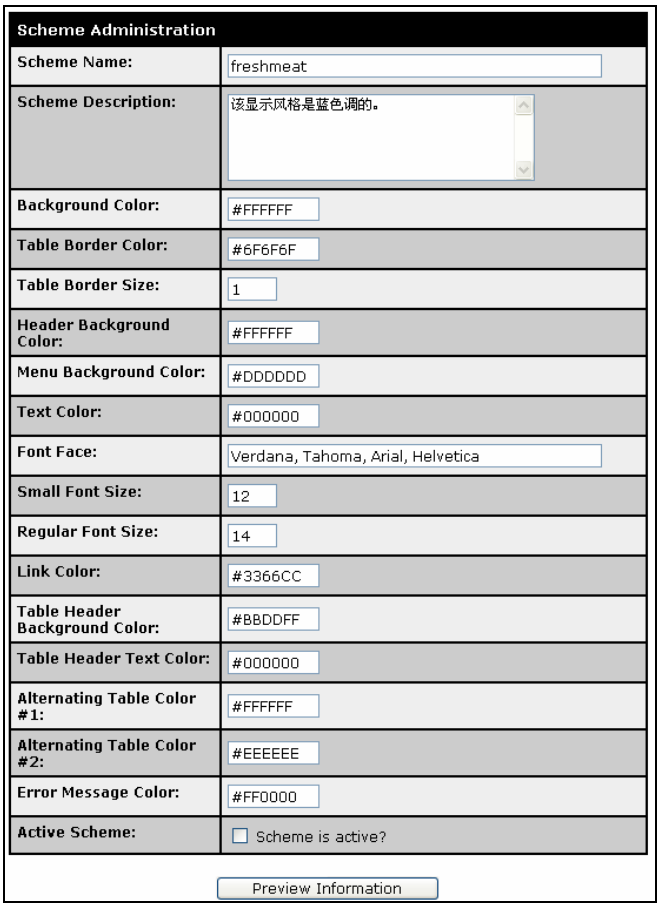


图 18-27　新增、编辑显示风格



图 18-28　显示风格预览

单击“Submit Scheme”按钮可以提交。

系统设置是由文件 scheme_admin.php 实现的，由于篇幅限制，代码参见光盘。

3．论坛管理

论坛管理是管理功能中最重要的部分。通过该功能，管理员能够编辑、删除以及添加论坛版块并设置各个论坛的基本属性，如论坛名称、论坛说明等。

在管理员菜单中单击“论坛管理”链接，可以查看现有的论坛信息，如图 18-29 所示。



图 18-29　查看现有论坛信息

单击“add new forum”链接可以添加新的论坛、单击“Edit”可以编辑现有论坛、单击“Delete”可以删除现有论坛。

添加新论坛的界面如图 18-30 所示。



图 18-30　添加新论坛

单击“Preview Information”按钮，则可以将更改保存到数据库，回到论坛列表查看状态。

其中“Forum Order”表示各个论坛的排列次序，如果想突出一个重点论坛，可以将其设置为 0。

新增论坛后的界面如图 18-31 所示。



图 18-31　新增论坛后的界面

论坛管理是由文件 forum_admin.php 实现的，如代码 18-9 所示。

【代码 18-9】　forum_admin.php

```
    <?php
    //省略部分代码
    /*按照以下步骤来创建新论坛版块 */
    switch ($step)
      {
        /* 显示论坛列表 */
        default:
        case 1:
          ShowForums();
          break;
        /* 显示新增论坛表单*/
        case 2:
          ShowForumForm( $forum_name, $forum_desc, $forum_order, $forum_id,
$type );
          break;
        /* 显示新增论坛预览信息*/
        case 3:
          echo "          <FORM action=\"?pid=forum_admin\" method=\"POST\"
name=\"forum_admin\">\n"
             . "            <TABLE cellpadding=\"5\" cellspacing=\"0\"
width=\"100%\" border class=\"table_border\">\n"
             . "              <TR>\n"
             . "                <TD class=\"table_header\" colspan=\"2\">Forum
Preview</TD>\n"
             . "              </TR>\n"
             . "              <TR bgcolor=\"" . TABLE_COLOR_1 . "\"
class=\"regular_text\">\n"
             . "                <TD width=\"25%\" valign=\"top\"><B>Forum
Name:</B></TD>\n"
             . "                <TD width=\"50%\">\n"
             . "                  $forum_name\n"
             . "                  <INPUT type=\"hidden\" name=\"forum_name\"
value=\"$forum_name\">\n"
             . "                </TD>\n"
             . "              </TR>\n"
             . "              <TR bgcolor=\"" . TABLE_COLOR_2 . "\"
class=\"regular_text\">\n"
             . "                <TD width=\"25%\" valign=\"top\"><B>Forum
```

```
Description:</B></TD>\n"
            . "                <TD width=\"50%\">\n"
            . "                  $forum_desc\n"
            . "                  <INPUT type=\"hidden\" name=\"forum_desc\"
value=\"$forum_desc\">\n"
            . "                </TD>\n"
            . "              </TR>\n"
            . "              <TR bgcolor=\"" . TABLE_COLOR_1 . "\"
class=\"regular_text\">\n"
            . "                <TD width=\"25%\" valign=\"top\"><B>Forum
Order:</B></TD>\n"
            . "                <TD width=\"50%\">\n"
            . "                  $forum_order\n"
            . "                  <INPUT type=\"hidden\" name=\"forum_order\"
value=\"$forum_order\">\n"
            . "                </TD>\n"
            . "              </TR>\n"
            . "            </TABLE>\n"
            . "            <INPUT type=\"hidden\" name=\"forum_id\"
value=\"$forum_id\">\n"
            . "            <INPUT type=\"hidden\" name=\"type\"
value=\"$type\">\n"
            . "            <INPUT type=\"hidden\" name=\"old_name\"
value=\"$old_name\">\n"
            . "            <CENTER>\n"
            . "              <BR>\n"
            . "              <INPUT type=\"submit\" value=\"Edit Forum\"
name=\"action\">\n"
            . "               \n"
            . "              <INPUT type=\"submit\" value=\"Submit Forum\"
name=\"action\">\n"
            . "            </CENTER>\n"
            . "          </FORM>\n";
         break;
       /* 添加到数据库 */
       case 4:
         /* 判断是否通过 POST 方法提交*/
         if ( $REQUEST_METHOD == "POST" )
           {
             /* 设置错误编码为 0*/
             $no_err = 0;
             /* 判断论坛名称是否有重复 */
             $SQL    = "SELECT COUNT(*) as forum_exists FROM " . TABLE_PREFIX .
"forums WHERE forum_name='$forum_name';";
             $results = ExeSQL($SQL);
             /* 如果已经存在，则提示已经存在*/
             while ($row = mysql_fetch_array($results))
               {
                 if ($row["forum_exists"] != 0)
                   {
                     echo "          <CENTER class=\"error_message\">A forum by
that name already exists!</CENTER><BR>\n";
```

```
                    $no_err++;
                  }
              }
            /* 如果没有 */
            if ($no_err == 0)
              {
                /* 保存进数据库*/
                $SQL    = "INSERT INTO " . TABLE_PREFIX . "forums (forum_name,
forum_desc, forum_order) VALUES ('$forum_name', '$forum_desc',
'$forum_order');";
                $results = ExeSQL($SQL);
                echo "          <CENTER class=\"normal_message\">The new forum
has successfully been added!</CENTER><BR>\n";
                ShowForums();
                return;
              }
            else
              ShowForumForm( $forum_name, $forum_desc, $forum_order, $forum_id,
$type );
          }
         else
          {
            /* 如果不是以 POST 方法提交，则显示添加新论坛界面 */
            echo "          <CENTER class=\"error_message\">Malformed request
detected!</CENTER><BR>\n";
            ShowForumForm( $forum_name, $forum_desc, $forum_order, $forum_id,
$type );
          }
         break;
       /* 编辑论坛信息，如果新论坛和旧论坛名称一样 */
       case 5:
        //省略部分代码
         /* 删除论坛 */
      case 6:
        //省略部分代码
     }
    //省略部分代码
    ?>
```

4．用户管理

用户管理是指管理员能够查看所有会员的注册信息，同时能够删除、添加以及编辑会员信息，还能够给用户进行授权。细心的读者会发现，在前面已经创建的两个论坛里，版主的位置都是空缺的，这是因为管理员还没有进行授权。

在管理员菜单中单击“用户管理”可以进入用户管理界面，如图 18-32 所示。



图 18-32　会员查询界面

为了方便管理员管理账号，系统提供了查询功能，如果用户输入“*”，则表示查询所有

的注册会员，如图 18-33 所示。

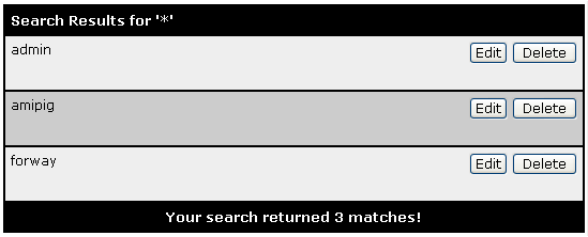


图 18-33　会员列表

如果想删除一个会员，只要单击“Delete”即可，或者可以单击“Edit”按钮进行编辑。编辑用户的界面和新用户注册的界面基本类似，这里只截取管理员进行授权的界面，如图 18-34 所示。

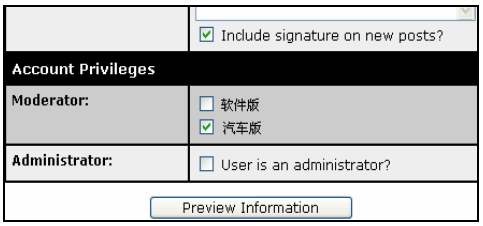


图 18-34　管理员授权

在图 18-34 中，管理员可以对其他用户进行授权，比如在“Moderator”一栏中显示的是所有的论坛名称，如果选中前面的复选框，则表示授权其为该论坛的版主。同时，也可以授权某个用户为系统管理员，但是要慎重，因为这样该用户就具备了管理员的所有权限。



当一个用户被授权做版主之后，能够管理处理本版内的文章，比如删除、编辑等。

编辑完成后首先进行预览，然后提交到数据库。这里授权“amipig”为汽车版的版主。“amipig”会出现在汽车论坛版主的位置，如图 18-35 所示。

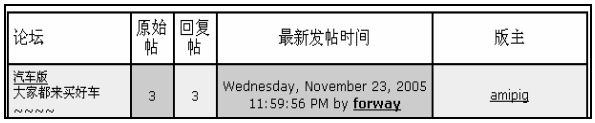


图 18-35　amipig 被授权做版主

读者可以试着用“amipig”进行登录，在登录到汽车版论坛的时候，会发现显示的时候多出“删除文章”的按钮。但是如果进入其他版，如软件版，则不会显示这些按钮。

用户管理主要由文件 user_admin.php 文件实现，代码参见光盘。

18.3.8　导航文件（index.php）

该文件主要进行功能导航，同时进行用户权限的验证。为了保证安全性，本系统的所有功能处理都必须通过 index.php 文件自动导航来实现。该文件代码比较长，可参考光盘。

以上介绍了整个 BBS 系统的核心功能的实现，该系统的功能以及流程均比较复杂，读者需要多花一些时间来配合光盘代码进行学习。

第 19 章　电子邮件收发系统

电子邮件的开发技术是 PHP 程序员必须掌握的技术之一。比如在注册系统中，可以通过邮件来发送激活码和用户密码。这些都无法利用现在其他的免费电子邮件系统来完成，必须将邮件发送的模块集成在自己的代码中实现。

在 PHP 中，进行电子邮件开发的方法比较多，通常采用 PHP 的 Mail 函数发送邮件，并利用 IMAP 库以及 PEAR 等技术。虽然通过 PEAR 可以开发出功能强大的电子邮件系统，但是需要单独安装 PEAR 包。

对于非商用的一些电子邮件处理，直接通过 PHP 的 Mail 函数来实现最为方便。但是 Mail 函数有一个最大的缺点就是不能进行 SMTP 的认证。因此，通过 Mail 函数发送邮件，最好能够在服务器安装一个 SMTP 的服务器，来转发邮件。

19.1　Mail 函数的简单应用

在 PHP 中，Mail 函数的声明如下所示：

```
    bool mail ( string to, string subject, string message [, string
additional_headers [, string additional_parameters]] )
```

下面介绍其中的参数。

to：收件人的电子邮件。可以使用以下几种形式：

```
    user@example.com
    user@example.com, anotheruser@example.com
    User <user@example.com>
    User <user@example.com>, Another User <anotheruser@example.com>
```

subject：信件的主题。

message：信件的内容。

additional_headers：可选参数，用来将一些信息插入到 Email 的头部，如发信人信息、抄送地址、密送地址等。如果添加多个信息，每个信息之间需要用“\r\n”隔开。

通过 Mail 函数发送电子邮件，需要在 php.ini 文件中进行 SMTP 邮件服务器地址和端口的设置。如下所示：

```
    [mail function]
    ; For Win32 only.
    SMTP = localhost
    smtp_port = 25
```

SMTP 为要设置的发送邮件的地址。通常免费的 SMTP，如 smtp.163.com、smtp.sohu.com 等都需要验证，因此用户不能设置该地址。尽量在自己的服务器上面安装一个 SMTP 服务器，如 IIS 等。

代码 19-1 所示添加了发送 Email 头的信息。

【代码 19-1】 additional_headers.php

```
<?php
$to     = 'nobody@example.com';
$subject = 'the subject';
$message = 'hello';
$headers = 'From: webmaster@example.com' . "\r\n" .
    'Reply-To: webmaster@example.com' . "\r\n" .
    'X-Mailer: PHP/' . phpversion();
mail($to, $subject, $message, $headers);
?>
```

代码 19-2 所示为一个邮件发送的简单实例。

【代码 19-2】 simplemail.php

```
<?php
// 是否提交邮件
  if ($mode == "submit") {   //是否以 html 方式
    if ($enable_html != "on") {
      //调用 htmlspecialchars 函数显示所有 html 标记
      $mail_body = htmlspecialchars ($mail_body);
    }
//  设置邮件头
    $headers .= "MIME-Version: 1.0 \n";
    $headers .= "Content-type: text/html; charset=iso-8859-1 \n";
    $headers .= "from:$mail_from\r\nCc:$mail_cc\r\nBcc:$mail_bcc";
//调用 mail 函数发送邮件
    if (mail ($mail_to, $mail_subject, $mail_body, $headers)) {
      print ("<h1><font color=\"#004000\">The e-mail was sent
successfully!</font></h1>");
    } else {
      print ("<h1><font color=\"#880000\">An error occurred while sending the
e-mail!</font></h1>");
    }
    exit;
  }
?>
<html>
<head>
<title>Send e-mail</title>
<script language="javascript">
  function DoSubmit ()
  {
/*
检测每个字段是否正确
*/
    if (document.form.mail_from.value == "") {
      alert ("You forgot to enter the 'from' field.");
      document.form.mail_from.focus ();
      return "";
    }
    if (document.form.mail_to.value == "") {
      alert ("You forgot to enter the 'to' field.");
```

```
        document.form.mail_to.focus ();
        return "";
      }
      if (document.form.mail_subject.value == "") {
        alert ("You forgot to enter the 'subject' field.");
        document.form.mail_subject.focus ();
        return "";
      }
      if (document.form.mail_body.value == "") {
        alert ("You forgot to enter the 'body' field.");
        document.form.mail_body.focus ();
        return "";
      }
      document.form.submit ();
    }
  </script>
  </head>
  <body>
  <form action="<?php print ($PHP_SELF); ?>" method="post" name="form">
  <!—省略表单显示代码-->
  </form>
  </body>
  </html>
```

执行界面如图 19-1 所示。



图 19-1　简单邮件发送系统

19.2　系统整体设计

前面介绍了 PHP 自带的 Mail 函数的简单应用。接下来将会介绍一个功能比较完善的电子邮件收发系统。该系统具备如下功能：

（1）通过 POP3 收取用户各种邮箱的邮件；

（2）在线阅读邮件；

（3）接收保存邮件内容；

（4）接收保存附件；

（5）在线回复邮件；

（6）在线处理邮件；
（7）发送普通邮件；
（8）发送多个附件；
（9）进行 SMTP 验证；
（10）通讯录管理；
（11）系统设置。


提示 由于发送电子邮件的 SMTP 协议和 POP3 协议内容比较多，本书将会以电子版的形式将这部分内容的介绍放在光盘中，包括 SMTP 协议和 POP3 协议的基础知识、客户端与邮件服务器的对话过程、邮件发送和收取的步骤等。

根据系统要实现的功能，得到系统的主要文件如下。
（1）index.php，系统登录界面。
（2）msglist.php，显示邮件列表。
（3）readmsg.php，用于阅读邮件内容。
（4）newmsg.php，用于发送新邮件。
（5）badlogin.php，用于显示登录失败信息。
（6）class_mime_decode.php，用于对邮件进行 MINE 编码。
（7）class_pop3_session.php，用于与服务器的 POP3 会话。
（8）class_smtp.php，用于与服务器进行 SMTP 会话。
（9）commom_functions.php，用于声明一些公用的函数和变量。
（10）config.php，该文件进行系统的初始化设置，如 SMTP 服务器，POP3 服务器等。
（11）download.php，用于下载邮件的附件。
（12）error.php，用于显示文件的错误信息。
（13）folders.php，用于创建邮箱文件夹。
（14）headers.php，用于显示页面的头部信息。
（15）logout.php，用于注销等。
（16）preferences.php，用于系统设置。
（17）search.php，用于邮件搜索。
（18）session_management.php，用于会话管理，主要进行登录用户的权限验证。
（19）upload.php，用于附件的上载。
（20）addressbook.php，用于通讯录的管理。

本系统采用后台代码和前台代码分离的技术，因此除了上面列出的后台 PHP 代码文件之外，还有许多前台的 HTML 文件，这些文件位于“themes\chinese”目录下。这样做的目的是可以为系统提供多版本的界面，而核心的后台代码不用任何的改变。比如本系统提供了中文界面和英文界面，用户可以根据需要选择不同的界面。

前台代码主要是 HTML 表单显示，这里就不列出其文件名。读者可以参考光盘对应目录下的文件。

19.3 系 统 实 现

系统下载后的邮件，会自动保存在对应目录下，而不是存储在数据库中，因此不需要创

建任何数据库，可以直接运行。

由于本系统代码量非常大，此处只能介绍一些核心的功能和模块，其他内容读者可以根据本书前面介绍的基础知识和光盘的源代码进行学习。

19.3.1　系统设置文件 config.php

该文件进行系统通用属性的设置，如代码 19-3 所示。

【代码 19-3】　config.php

```
    <?
    /*****************************************************
    Defaults:
    1 - Yes/On/True
    0 - No/Off/False
    */
    //设置临时目录
    $temporary_directory = "temporary_files/";
    //SMTP 服务器地址
    $smtp_server = "localhost";
    //文件的最大限制
    $quota_limit = 2048;  //2048 Kb
    //pop3 服务器设置，可以设置多个
    //$pop3_servers[0]["domain"] = "1stDomain.com";
    $pop3_servers[0]["server"] = "pop3.163.com";
    // 2nd
    //$pop3_servers[1]["domain"] = "2ndDomain.com";
    //$pop3_servers[1]["server"] = "pop3.2ndDomain.com";
    //选择语言版本
    $default_language    = 1;
    $allow_user_change   = 1; //允许用户选择语言版本
    $languages[0]["name"] = "中文";
    $languages[0]["path"] = "themes/chinese"; // 不同的版本，文件目录不同
    $languages[1]["name"] = "English";
    $languages[1]["path"] = "themes/english"; //
    //设置时间显示格式
    $date_format = "d/m/y h:m"; // d = day, m = month, y = year, h = hour, m =
minutes
    //是否允许发送 RETR 后删除原来邮件
    $pop_use_top = 1;
    //是否允许 html 格式的邮件内容
    $mime_show_html = 1;
    //SMTP 是否需要验证
    $use_password_for_smtp = 1;
    //版本信息
    $appversion = "2.0";
    $appname = "forway mail";
    //发送邮件时自动添加说明内容
    $footer = "
    _____________________________________________________
    This mail was sent by $appname $appversion
    ";
```

```
//允许调试
$enable_debug = 0 ;
//会话失效时间
$idle_timeout = 10; //分钟
//邮件的排序
$default_sortby = "date";
$default_sortorder = "DESC";
$send_to_trash_default = 1; //默认删除的邮件被放入垃圾箱
$st_only_ready_default = 1; //只读邮件，否则删除
$save_to_sent_default  = 1; //将发送完毕的邮件放入 sent 文件夹
$empty_trash_default   = 1; //退出时清空垃圾箱
$sortby_default       = "date"; //默认按照日期排序 ，可以按照大小、发件人等排序
$sortorder_default    = "DESC"; //默认降序排序
$rpp_default          = 10; // 每页默认显示邮件数量为 10，可选的为: 10,20,30,40,50
$add_signature_default = 0; //是否自动添加签名
$signature_default    = ""; // 默认签名为空
?>
```

19.3.2 POP3 系统登录

当用户在浏览器中输入“http://localhost/mail_2/index.php”时，就会自动进入系统登录界面，如图 19-2 所示。



图 19-2 系统登录界面

用户可以像登录其他的电子邮箱一样登录该系统。其中用户名和密码为用户在邮件系统注册的用户名和密码。



注意 要和在 config.php 中设置的 POP3 服务器对应。比如设置了 pop3.163.com，则只能登录 163 的邮箱。如果设置了 pop3.zj.com，则只能登录 www.zj.com 的邮箱。

下面将 config.php 文件的 POP3 服务器设置成 pop3.zj.com，然后登录该邮件系统。登录成功后的界面如图 19-3 所示。

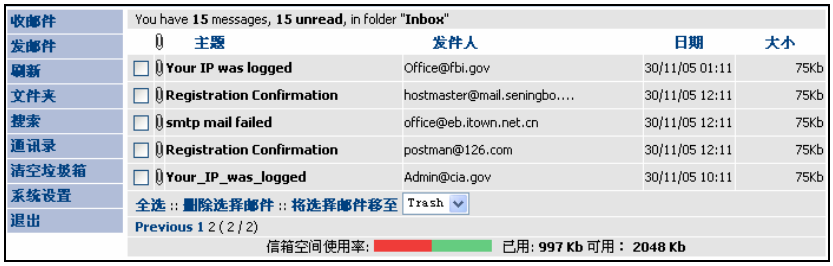


图 19-3 收件箱

图 19-3 所示显示了邮箱中所有的邮件，可以查看到的信息有总邮件数量、未读邮件数量、

邮件的发件人、主题、日期、大小和是否有附件等。

下面介绍邮箱登录功能的实现。其中邮箱的用户名、密码验证以及获取邮件信息的部分全部在 msglist.php 中实现。

首先介绍 index.php 和 login.html 两个文件，因为这两个文件显示了登录界面。其中 index.php 全部为后台的 PHP 代码，而 login.htm 为前台的 HTML 代码。

首先看 login.htm 的代码，如代码 19-4 所示。

【代码 19-4】　login.htm

```
    <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
    <html>
    <head>
        <title>UebiMiau - Login</title>
    </head>
    <link rel="stylesheet" href="themes/english/webmail.css" type="text/css">
    <!--%UM_JS%-->
    <body bgcolor="#004080" text="#FFFFFF" link="#FFFFFF" vlink="#FFFFFF" alink=
"#FFFFFF">
    <br><br><br>
    <table width="400" border="0" cellspacing="0" cellpadding="2"
align="center">
        <tr>
            <td><img src="images/logo.gif" width="183" height="73" border="0"
alt=""></td>
        <tr>
            <td bgcolor=white>
                <table width="100%" border="0" cellspacing="1" cellpadding="1"
align="center">
                <form name="form1" action="msglist.php" method=post>
                    <input type="hidden" name=sid value="<!--%UM_SID%-->">
                    <tr><td align=right class="title" colspan=2>.: <b>登录邮件系
统</b> :.</td>
                    <!--%UM_VARIABLE_SERVER_BEGIN%-->
                    <tr><td align=right class="right"><b>Email</b>:  </td><td
class="default"><input type=text size=25 name=f_email value="" class="textbox"></td>
                    <tr><td align=right class="right"><b>用户名</b>:
 </td><td class="default"><input type=text size=15 name=f_user
value="<!--%UM_F_USER%-->" class="textbox"></td>
                    <tr><td align=right class="right"><b>密码</b>:  </td><td
class="default"><input type=password size=15 name="f_pass" class="textbox"></td>
                    <tr><td align=right class="right"><b>服务器</b>:  </td><td
class="default"><input type=text size=20 name="f_server" class="textbox"></td>
                    <!--%UM_VARIABLE_SERVER_END%-->
                    <!--%UM_STATIC_SERVER_BEGIN%-->
                    <tr><td align=right class="right"><b>用户名</b>:  </td><td
class="default"><input type=text size=15 name=f_user value="<!--%UM_F_USER%-->"
class="textbox">  <b><!--%UM_SERVERS%--></b></td>
                    <tr><td align=right class="right"><b>密码</b>:  </td><td
class="default"><input type=password size=15 name="f_pass" class="textbox"></td>
                    <!--%UM_STATIC_SERVER_END%-->
```

```
                    <!--%UM_LANGUAGE_BEGIN%-->
                    <tr><td align=right class="right"><b>选择语言</b>:  
</td><td class="default"><!--%UM_LANGUAGES%--></td>
                    <!--%UM_LANGUAGE_END%-->
                    <tr><td align=right> </td><td><input type=submit name=
submit value="登录 >>"></td>
                </form>
                </table>
            </td>
        </tr>
    </td>
    </table>
    </body>
    </html>
```

index.php 如代码 19-5 所示。

【代码 19-5】 index.php

```
    <?php
    include("config.php");
    include("commom_functions.php");
    error_reporting (E_ALL ^ E_NOTICE);
    $sid = md5(uniqid(""));
    //获取登录显示界面的模板
    //$login_template 在 common_functions.php 中定义
    //其中每个 php 文件都对应着一个 html 文件，一个用于后台一个用于前台
    //登录界面中的 html 前台模板是$lngpath/login.htm
    //变量$lngpath 根据用户选定的语言版本来设定，在 common_functions.php 中定义
    //如果用户选择的是中文界面，则路径为 themes/chinese
    //登录界面中模板文件为 themes/chinese/login.htm
    //首先读入登录模板文件，此时$tcontent 保存着 login.html 的文件内容
    $tcontent = read_file($login_template);
    //定义变量$jssource，保存 javascript 的代码
    $jssource = "
    <script language=javascript>
    function select_language(lid) {
        location = 'index.php?lid='+lid+'&f_user='+escape(document.forms[0].
f_user.value)
    }
    </script>
    ";
    //接着进行替换，将登录模板文件中定义的一些特殊字符串替换成目标字符串
    $tcontent    = eregi_replace("<!--%UM_JS%-->",$jssource,$tcontent);
    $tcontent    = eregi_replace("<!--%UM_SID%-->",strtoupper($sid),$tcontent);
    $tcontent    = eregi_replace("<!--%UM_F_USER%-->",htmlspecialchars($f_user),
$tcontent);
    $startvar = strpos($tcontent,"<!--%UM_VARIABLE_SERVER_BEGIN%-->");
    $endvar = strpos($tcontent,"<!--%UM_VARIABLE_SERVER_END%-->")+31;
    $startstat = strpos($tcontent,"<!--%UM_STATIC_SERVER_BEGIN%-->");
    $endstat = strpos($tcontent,"<!--%UM_STATIC_SERVER_END%-->")+29;
    $aval_servers = count($pop3_servers);  //获得服务器名称
    if($aval_servers != 0) {
```

```
        // 修正服务器名称
        $correct = substr($tcontent,$startstat+31,$endstat-$startstat-60);
        if ($aval_servers == 1) {
            $tmp = "@".$pop3_servers[0]["domain"]." <input type=hidden name=six
value=0>";
            $correct = eregi_replace("<!--%UM_SERVERS%-->",$tmp,$correct);
        } else {
            $tmp = "<select name=six>\r";
            for($i=0;$i<$aval_servers;$i++)
                $tmp .= "<option value=$i>@".$pop3_servers[$i]["domain"]."\r";
            $tmp .= "</select>\r";
            $correct = eregi_replace("<!--%UM_SERVERS%-->",$tmp,$correct);
        }
    } else {
        $correct = substr($tcontent,$startvar+33,$endvar-$startvar-64);
        //variable server
    }
    $tcontent = substr($tcontent,0,$startvar).$correct.substr($tcontent,$endstat);
    $avallangs = count($languages);
    if($avallangs == 0) die("You must provide at least one language");
    $startlng = strpos($tcontent,"<!--%UM_LANGUAGE_BEGIN%-->");
    $endlng = strpos($tcontent,"<!--%UM_LANGUAGE_END%-->")+24;
    if($allow_user_change) {
        $cleantext = substr($tcontent,$startlng+26,$endlng-$startlng-50);
        $langsel = "<select name=lng onChange=\"select_language(this.options
[this.selectedIndex].value)\">\r";
        $def_lng = (is_numeric($lid))?$lid:$default_language;
        for($i=0;$i<$avallangs;$i++) {
            $selected = ($lid == $i)?" selected":"";
            $langsel .= "<option value=$i$selected>".$languages[$i]["name"]."\r";
        }
        $langsel .= "</select>\r";
        $cleantext = eregi_replace("<!--%UM_LANGUAGES%-->",$langsel,$cleantext);
        $tcontent = substr($tcontent,0,$startlng).$cleantext.substr($tcontent,
$endlng);
    } else
        $tcontent = substr($tcontent,0,$startlng).substr($tcontent,$endlng);
    echo($tcontent);
    ?>
```

以上代码首先读入 login.htm 的文件代码，然后根据实际情况设置文件代码中的关键字“<!--%UM_JS%-->”、同时根据需要获取“<!--%UM_VARIABLE_SERVER_BEGIN%-->”和“<!--%UM_VARIABLE_SERVER_END%-->”等类似标记对的内容，动态的输出 login.htm 文件的内容。

19.3.3　收取邮件和附件

进行 POP3 身份验证以及邮件的获取都是通过 msglist.php 文件实现的。进行服务器连接、邮件收取以及附件保存都是通过一个核心的类，即 pop3_session 类来实现的，学习邮件收发系统的核心就是学习该类，该类保存在 class_pop3_session.php 文件中，如代码 19-6 所示。

【代码 19-6】 class_pop3_session.php

```
    <?
    class pop3_session {  //该类用于收取邮件
        var $pop_connection,
            $pop_port,  //端口
            $pop_error_msg,  //错误信息
            $pop_user,  //用户名
            $pop_pass,  //密码
            $pop_server;  //服务器地址
        function pop_get_line() {  //获取服务器响应
            global $enable_debug;
            $buffer = fgets($this->pop_connection,10240);
            $buffer = eregi_replace("\r","",$buffer);
            $buffer = eregi_replace("\n","",$buffer);
            if($enable_debug) {
                $output = (eregi("^(\+OK)|^(\-ERR)",$buffer))?"<- <b>".htmlspecia-
lchars($buffer)."</b>":htmlspecialchars($buffer);
                echo("<font style=\"font-size:12px; font-family: Courier New;
background-color: white; color: black;\"> $output</font><br>\r\n");flush();
            }
            return $buffer;
        }
        function pop_send_command($cmd) {  //发送命令到服务器
            global $enable_debug;
            if($this->pop_connection) {
                if($enable_debug) {
                    $output = (eregi("^(PASS)",$cmd))?"PASS ****":$cmd;
                   echo("<font style=\"font-size:12px; font-family: Courier New;
background-color: white; color: black;\">-&gt;
<em><b>".htmlspecialchars($output)."</b></em></font><br>\r\n");flush();
                }
                fwrite($this->pop_connection,"$cmd\r\n");
                return 1;
            }
            return 0;
        }
        function pop_connect() {  //调用 fsockopen 命令连接邮件服务器
            $this->pop_connection = fsockopen($this->pop_server, $this->pop_
port, $errno, $errstr, 60);
            if($this->pop_connection) {
                $buffer = $this->pop_get_line();
                if(ereg("^(\+OK)",$buffer)) return 1;
                else return 0;
            }
            return 0;
        }
        function pop_auth() {  //服务器用户名和密码验证
            global $userfolder,$error_permiss,$temporary_directory,$idle_timeout;
            if($this->pop_connection) {
            $this->pop_send_command("USER ".$this->pop_user);  // 发送 user 命令
```

```
                $buffer = $this->pop_get_line();
                if(ereg("^(\+OK)",$buffer)) {
                $this->pop_send_command("PASS ".$this->pop_pass);  //发送密码命令
                    $buffer = $this->pop_get_line();

                    if(ereg("^(\+OK)",$buffer)) {
                        if(!file_exists($userfolder))
                            if(!@mkdir($userfolder,0777)) die("<h1><br><br><br>
<center>$error_permiss</center></h1>");
                            if(!file_exists($userfolder."inbox"))
                                mkdir($userfolder."inbox",0777);
                            if(!file_exists($userfolder."trash"))
                                mkdir($userfolder."trash",0777);
                            if(!file_exists($userfolder."sent"))
                                mkdir($userfolder."sent",0777);
                            if(!file_exists($userfolder."sent"))
                                mkdir($userfolder."sent",0777);
                            if(!file_exists($userfolder."_attachments"))
                                mkdir($userfolder."_attachments",0777);
                            if(!file_exists($userfolder."_infos"))
                                mkdir($userfolder."_infos",0777);
                            $sessiondir = $temporary_directory."_sessions/";
                            // Clean old sessions
                            $all=opendir($sessiondir);
                            while ($file=readdir($all)) {
                                $thisfile = $sessiondir.$file;
                                if (is_file($thisfile)) {
                                $idle = intval((time()-@filemtime($thisfile))/60);
                                    if(($idle_timeout+10) < $idle)
                                        @unlink($thisfile);
                                }
                            }
                            closedir($all);
                            unset($all);

                            return 1;
                    }
                    else { $this->pop_error_msg = $buffer; return 0; }
                } else return 0;
            }
            return 0;
        }
    function pop_retr_msg($mnum,$mid,$msize=10000,$check=1) {  //获取邮件详细信息
            global $pop_use_top,$userfolder,$appname,$appversion,$error_retrieving;
            $md = new mime_decode();
            if($check) {
                $this->pop_send_command("TOP $mnum 0");
                $buffer = $this->pop_get_line();
                if(ereg("^(\+OK)",$buffer)) {
                    unset($header);
                    while (!feof($this->pop_connection)) {
                        $buffer = $this->pop_get_line();
```

```
                    if(trim($buffer) == ".") break;
                    if(strlen($buffer) > 3)
                        $header .= "$buffer\r\n";
                }
                $mail_info = $md->get_mail_info($header);
                $mailmid = $mail_info["message-id"];
                if($mid != md5($mailmid)) {
                    $this->pop_error_msg = $error_retrieving;
                    return 0;
                }
            } else return 0;
        }
        $command = ($pop_use_top)?"TOP $mnum $msize":"RETR $mnum";
        $this->pop_send_command($command);
        $buffer = $this->pop_get_line();
    if(!ereg("^(\+OK)",$buffer)){$this->pop_error_msg=$buffer;return 0;}
        while (!feof($this->pop_connection)) {
            $buffer = str_replace("\n","",$this->pop_get_line());
            if(trim($buffer) == ".") break;
            $msg .= "$buffer\r\n";
        }
        $msg = "X-Decoded-By: $appname $appversion\r\n".$msg;
        $parts = $md->fetch_structure($msg);
        $header = $parts["header"];
        $mail_info = $md->get_mail_info($header);
        $flocalname = $userfolder."inbox/".md5($mail_info["subject"]. $mail_
info["message-id"].$mail_info["date"]).".eml";
        $tmpfile = fopen($flocalname,"wb+");
        fwrite($tmpfile,$msg);
        fclose($tmpfile);
        return $msg;
    }
    function pop_dele_msg($mnum,$mid,$msize,$send_to_trash = 1) {  //删除邮件
        global $userfolder,$error_retrieving;
        $this->pop_send_command("TOP $mnum 0");
        $buffer = $this->pop_get_line();
        if(ereg("^(\+OK)",$buffer)) {
            unset($header);
            while (!feof($this->pop_connection)) {
                $buffer = $this->pop_get_line();
                if(trim($buffer) == ".") break;
                if(strlen($buffer) > 3)
                    $header .= "$buffer\r\n";
            }
            $md = new mime_decode();
            $mail_info = $md->get_mail_info($header);
            if($mid != md5($mail_info["message-id"])) {
                $this->pop_error_msg = $error_retrieving;
                return 0;
            }
            $filename = md5($mail_info["subject"].$mail_info["message-id"].
$mail_info["date"]).".eml";
```

```
            $flocalname = $userfolder."inbox/$filename";
            $flocaltrashname = $userfolder."trash/$filename";
            if ($send_to_trash && !file_exists($flocalname)) {
                $tempmail = $this->pop_retr_msg($mnum,$mid,$msize,0);
                $tempmail = $md->set_as($tempmail,0);
                $tmpfile = fopen($flocalname,"wb+");
                fwrite($tmpfile,$tempmail);
                fclose($tmpfile);
            }
            if($send_to_trash)
                @copy($flocalname,$flocaltrashname);
            @unlink($flocalname);
            $this->pop_send_command("DELE $mnum");
            $buffer = $this->pop_get_line();
            if(!ereg("^(\+OK)",$buffer)) {
                $this->pop_error_msg = $buffer;
                return 0;
            } else return 1;
        }
        $this->pop_error_msg = $buffer;
        return 0;
    }
    function pop_list_msgs() {  //显示邮件列表
        global $userfolder,$use_progress_bar;
        $msglist = Array();
        $this->pop_send_command("LIST");
        unset($buffer);
        $buffer = $this->pop_get_line();
        if(ereg("^(\+OK)",$buffer)) {
            $counter = 0;
            while (!feof($this->pop_connection)) {
                $buffer = $this->pop_get_line();
                if(trim($buffer) == ".") break;
                $msgs = split(" ",$buffer);
                if(is_numeric($msgs[0])) {
                    $msglist[$counter]["id"] = $counter+1; //$msgs[0];
                    $msglist[$counter]["msg"] = $msgs[0];
                    $msglist[$counter]["size"] = $msgs[1];
                    $counter++;
                }
            }
            if(count($msglist) == 0) return $msglist;
            $md = new mime_decode();
            for($i=0;$i<count($msglist);$i++) {
                $this->pop_send_command("TOP ".$msglist[$i]["msg"]." 0");
                $buffer = $this->pop_get_line();
                if(ereg("^(\+OK)",$buffer)) {
                    while (!feof($this->pop_connection)) {
                        $buffer = $this->pop_get_line();
                        if(trim($buffer) == ".") break;
                        if(strlen($buffer) > 3)
                            $header .= "$buffer\r\n";
```

```
                        }
                        $mail_info = $md->get_mail_info($header);
                        $msglist[$i]["date"] = $mail_info["date"];
                        $msglist[$i]["subject"] = $mail_info["subject"];
                        $msglist[$i]["message-id"] = $mail_info["message-id"];
                        $msglist[$i]["from"] = $mail_info["from"];
                        $msglist[$i]["fromname"] = $mail_info["from"][0]["name"];
                        $msglist[$i]["to"] = $mail_info["to"];
                        $msglist[$i]["cc"] = $mail_info["cc"];
                        $msglist[$i]["headers"] = $header;
                        $msglist[$i]["attach"] = (eregi("(multipart/mixed|mult-
ipart/related|application)",$mail_info["content-type"]))?1:0;
                       $flocalname=$userfolder."inbox/".md5($mail_info["subject"].
$mail_info["message-id"].$mail_info["date"]).".eml";
                        $msglist[$i]["localname"] = $flocalname;
                        $msglist[$i]["read"] = file_exists($flocalname)?1:0;
                    }
                    $header = "";
                }
            }

            return $msglist;
        }
        function pop_disconnect() {  //断开与服务器的连接
            $this->pop_send_command("QUIT");
            $tmp = $this->pop_get_line();
            fclose($this->pop_connection);
            return 1;
        }
    }
    ?>
```

在该类中，提供了许多函数进行邮件接收处理，如连接服务器、进行用户名和密码的验证、获取邮件列表、阅读邮件、删除邮件、发送命令以及获取服务器相应信息等函数。核心是由 fsockopen 函数与服务器进行通信。下面简单介绍 POP3 接收邮件的过程。

POP 的全称是 Post Office Protoco ，即邮局协议，用于电子邮件的接收，现在常用的是第 3 版，简称为 POP3。通过 POP 协议，客户机登录到服务器后，可以对自己的邮件进行删除，或是下载到本地。下载后，电子邮件的客户软件就可以在本地对邮件进行修改、删除等。

POP 服务器一般使用的是 TCP 的 110 号端口，如果使用的是 Foxmail，在其收邮件的时候，可以看到其信息提示窗口有如下一些命令：

```
    "正在连接到  62.123.23.123:110";
    "USER BOSS_CH";
    "PASS.............";
```

下面来看一段与 POP3 服务器对话的实录:

```
    telenet pop.china.com 110
    +OK AIMC POP service (mail2.china.com) is ready.
    USER boss_ch
    +OK Please enter password for user .
    PASS ******
```

```
    +OK boss_ch has 1 messages (750 octets)
    STAT
    +OK 1 750
    LIST
    +OK 1 messages (750 octets)    1 750
    RETR 1
    +OK 750 octets
    Received: from smtp2.ptt.js.cn([202.102.24.37]) by china.com(JetMail
2.5.3.0)
       with SMTP id jm4839cc4227; Sat, 23 Sep 2000 05:31:21 –0000
    Received: from chenjunqing ([61.155.120.6]) by smtp2.ptt.js.cn
    (Netscape Messaging Server 4.15) with SMTP id G1BRHJ03.V07 for
    ; Sat, 23 Sep 2000 13:34:31 +0800
    Date: Sat, 23 Sep 2000 13:34:18 +0800
    From: =?ISO-8859-1?Q?=B3=C2=BF=A1=C7=E5?=
    To: boss_ch@china.com
    Subject: =?ISO-8859-1?Q?=D3=CA=BC=FE=CA=BE=C0=FD?=
    X-mailer: FoxMail 3.1 [cn]
    Mime-Version: 1.0
    Content-Type: text/plain; charset="GB2312"
    Content-Transfer-Encoding: 8bit
    Message-ID:     您好！这是一个邮件的小示例
    QUIT
    +OK Pop server at signing off.
```

以下对几个常用的 POP3 命令进行简单的介绍，如表 19-1 所示。

表 19-1　　POP3 命令

命　令	参　数	状　态	描　述
USER	username	认可	此命令与下面的 pass 命令若成功，将导致状态转换
PASS	password	认可	
APOP	Name,Digest	认可	Digest 是 MD5 消息摘要
STAT	None	处理	请求服务器发回关于邮箱的统计资料，如邮件总数和总字节数
UIDL	[Msg#]	处理	返回邮件的惟一标识符，POP3 会话的每个标识符都将是惟一的
LIST	[Msg#]	处理	返回邮件数量和每个邮件的大小
RETR	[Msg#]	处理	返回由参数标识的邮件的全部文本
DELE	[Msg#]	处理	服务器将邮件标记为删除，由 QUIT 命令执行
RSET	None	处理	服务器将重置所有标记为删除的邮件，用于撤消 DELE 命令
TOP	[Msg#]	处理	服务器将返回由参数标识的邮件前 n 行内容，n 必须是正整数
NOOP	None	处理	服务器返回一个肯定的响应，不做任何操作
QUIT	None	更新	退出

19.3.4　阅读邮件和保存附件

阅读邮件由 readmsg.php 文件实现。在邮件列表中，单击任何一封邮件的标题，即可阅读该邮件。如果该邮件有附件，还可以下载或者在浏览器中打开。阅读邮件的功能实现如图 19-4 所示。

在图 19-4 中可以单击“next”链接查看下一封邮件、单击“Reply”链接回复邮件、单击“Reply to all”链接回复全部、单击“Headers”链接查看邮件的头信息。另外可以直接删除或者将邮件移到垃圾箱等其他目录。



图 19-4　阅读邮件界面

单击“Headers”链接后，出现的邮件头信息如图 19-5 所示。

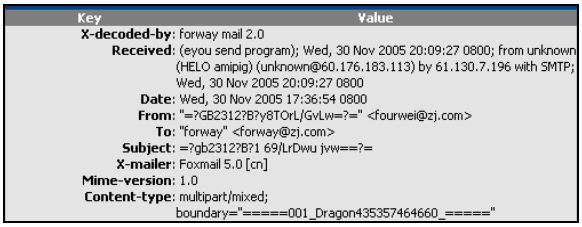


图 19-5　邮件头信息

如果单击邮件的附件，则会自动弹出“保存”对话框，如图 19-6 所示。如果浏览器能够识别该附件，则新打开浏览器窗口显示附件内容。

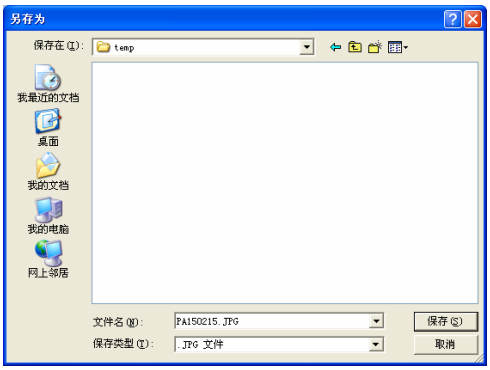


图 19-6　保存附件

提示　用户每阅读一封邮件后，系统会自动将邮件保存在指定的目录下面。比如本系统默认保存在当前目录的“temporary_files”目录下，则会为每个用户名添加一个目录。

图 19-7 所示为笔者的邮件保存目录。



图 19-7　temporary_files 的子目录

如果本次用“forway”登录，则相关的邮件信息则会自动保存在“forway_pop3.zj.com”目录下，打开该目录，如图 19-8 所示。

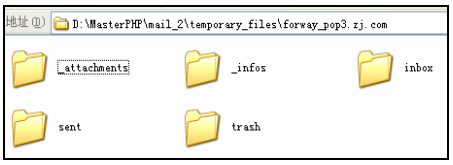


图 19-8　forway_pop3.zj.com 的子目录

其中_attachments 目录保存用户发送附件的附件内容、inbox 保存阅读过的邮件内容、sent 保存发送的邮件内容、trash 保存垃圾邮件。打开“inbox”目录，可以看到所有曾经阅读过的邮件。双击这些邮件，可以调用 Outlook 来查看新建的内容，如图 19-9 所示。

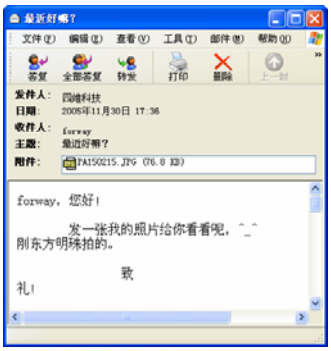


图 19-9　使用 Outlook 阅读保存的邮件

阅读邮件的核心代码在 readmsg.php 文件中，读者可参照光盘学习。

19.3.5　发送或者回复邮件

在功能菜单中单击“发邮件”或者直接在阅读邮件的时候单击“Reply”链接，即可进入发邮件的界面，如图 19-10 所示。

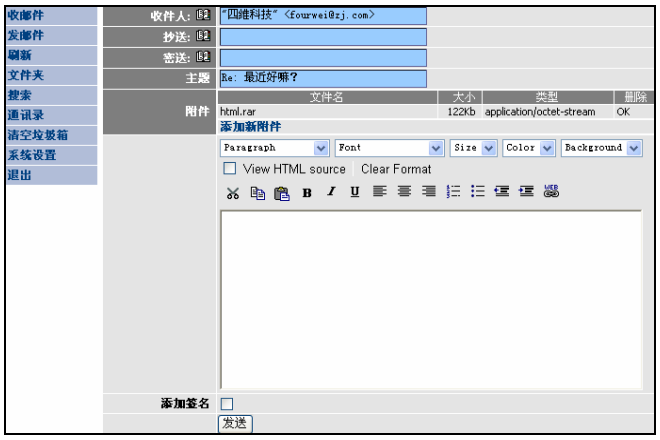


图 19-10　发邮件或者回复邮件

如果单击“发邮件”链接，则会显示一个空白的邮件发送界面。如果是回复邮件，则自动在收件人地址中填上原来发信的人的地址。同时自动填写邮件的主题，并保留原来邮件的

内容。在发送邮件的时候，可以添加多个附件。以上这些都是由 newmsg.php 文件来完成的。输入邮件内容后，单击“发送”按钮，可以发送邮件。发送成功后会显示发送成功的信息。

发送邮件，其核心是调用了一个 SMTP 发送类，该类的名称为 phpmailer，该类保存在 class_smtp.php 文件中。同时，phpmailer 类又引用了另外一个类 SMTP。

class_smtp.php 文件如代码 19-7 所示。

【代码 19-7】 class_smtp.php

```
<?php
class phpmailer {
    var $Priority       = 3;
    var $CharSet        = "iso-8859-1";
    var $ContentType     = "text/plain";
    var $Encoding       = "8bit";
    var $From           = "root@localhost";
    var $FromName       = "root";
    var $Subject        = "";
    var $Body           = "";
    var $WordWrap       = true;
    var $MailerDebug     = false;
    var $UseMSMailHeaders = true;
    // SMTP
    var $Host       = "localhost";  //服务器地址
    var $Port       = 25;//端口
    var $Helo       = "";  //发送 HELO 命令
    var $Timeout    = 10; // 超时时间限制
    var $SMTPDebug   = false;
    ////////////////////////////////////////////////
    // 常用变量声明
    ////////////////////////////////////////////////
    var $version      = "";
    var $to           = array();
    var $cc           = array();
    var $bcc          = array();
    var $ReplyTo      = array();
    var $attachment    = array();
    var $CustomHeader   = array();
    var $boundary      = false;
    var $ErrorAlerts    = Array();
    var $blUseAuthLogin = false;
    var $AuthUser      = "";  //smtp 用户名
    var $AuthPass    = "";  //smtp 密码，现在大部分的免费 smtp 都必须提供用户名和密码
    function UseAuthLogin($user,$pass) {  //smtp 变量名和密码
        $this->blUseAuthLogin = true;
        $this->AuthUser = $user;
        $this->AuthPass = $pass;
    }
    ////////////////////////////////////////////////
    // 是否是以 html 格式发送
    ////////////////////////////////////////////////
    function IsHTML($bool) {
```

```
            if($bool == true)
                $this->ContentType = "text/html";
            else
                $this->ContentType = "text/plain";
        }
        /////////////////////////////////////////////////
        // 显示版本信息
        /////////////////////////////////////////////////

        function Start() {
            global $appname,$appversion;
            $this->Version = $appname." ".$appversion;
            $this->Helo = ereg_replace("[^A-Za-z0-9]","",$appname);
        }
        /////////////////////////////////////////////////
        // 接收
        /////////////////////////////////////////////////
        function AddAddress($address, $name = "") {  //收件地址
            $cur = count($this->to);
            $this->to[$cur][0] = trim($address);
            $this->to[$cur][1] = $name;
        }
        function AddCC($address, $name = "") {   //抄送地址
            $cur = count($this->cc);
            $this->cc[$cur][0] = trim($address);
            $this->cc[$cur][1] = $name;
        }
        function AddBCC($address, $name = "") {  //密送地址
            $cur = count($this->bcc);
            $this->bcc[$cur][0] = trim($address);
            $this->bcc[$cur][1] = $name;
        }
        function AddReplyTo($address, $name = "") {  //回复地址
            $cur = count($this->ReplyTo);
            $this->ReplyTo[$cur][0] = trim($address);
            $this->ReplyTo[$cur][1] = $name;
        }
        /////////////////////////////////////////////////
        // 发送
        /////////////////////////////////////////////////
        /**
        * Send method creates message and assigns Mailer.  Returns bool.
        * @public
        * @returns bool
        */
        function Send() {   //创建邮件发送头
            if(count($this->to)+count($this->cc)+count($this->bcc) == 0) {
                $this->error_handler("You must provide at least one recipient
email address");
                return false;
            }
            $header = $this->create_header();
```

```
            if(($body = $this->create_body()) === false)
                return false;
            if($this->smtp_send($header, $body) === false)
                return false;
            return sprintf("%s%s", $header, $body);
        }
        function smtp_send($header, $body) {   //通过 smtp 发送邮件

            $smtp = new SMTP;  //创建一个 smtp 类
            $smtp->do_debug = $this->SMTPDebug;
            $hosts = explode(";", $this->Host);
            $index = 0;
            $connection = false;
            while($index < count($hosts) && $connection == false) {
                if($smtp->Connect($hosts[$index], $this->Port, $this->Timeout))
                    $connection = true;
                $index++;
            }
            if(!$connection) { //连接服务器
                $this->error_handler("SMTP Error: could not connect to SMTP host
server(s)");
                return false;
            }
            if($this->blUseAuthLogin) {  //进行 smtp 权限验证
        if(!$smtp->AuthHello($this->Helo,$this->AuthUser,$this->AuthPass)) {
                $this->error_handler("SMTP Error: Invalid username/password");
                    return false;
                }
            } else
                $smtp->Hello($this->Helo);  //发送 Helo 命令
            $smtp->MailFrom(sprintf("<%s>", $this->From));
            for($i = 0; $i < count($this->to); $i++)
                $smtp->Recipient(sprintf("<%s>", $this->to[$i][0]));
            for($i = 0; $i < count($this->cc); $i++)
                $smtp->Recipient(sprintf("<%s>", $this->cc[$i][0]));
            for($i = 0; $i < count($this->bcc); $i++)
                $smtp->Recipient(sprintf("<%s>", $this->bcc[$i][0]));
            if(!$smtp->Data(sprintf("%s%s", $header, $body))) {  //发送邮件
                $this->error_handler("SMTP Error: Data not accepted");
                return false;
            }
            $smtp->Quit();
        }
        ////////////////////////////////////////////////
        // 信息生成
        ////////////////////////////////////////////////
        function addr_append($type, $addr) {  //添加地址
            $addr_str = "";
            $addr_str .= sprintf("%s: %s <%s>", $type, $addr[0][1], $addr[0][0]);
            if(count($addr) > 1) {
                for($i = 1; $i < count($addr); $i++) {
                    $addr_str .= sprintf(", %s <%s>", $addr[$i][1], $addr[$i][0]);
```

```
                }
                $addr_str .= "\r\n";
            } else
                $addr_str .= "\r\n";
            return($addr_str);
        }
        function wordwrap($message, $length) {  //自动换行
            $line = explode("\n", $message);
            $message = "";
            for ($i=0 ;$i < count($line); $i++)
            {
                $line_part = explode(" ", trim($line[$i]));
                $buf = "";
                for ($e = 0; $e<count($line_part); $e++)
                {
                    $buf_o = $buf;
                    if ($e == 0)
                        $buf .= $line_part[$e];
                    else
                        $buf .= " " . $line_part[$e];
                    if (strlen($buf) > $length and $buf_o != "")
                    {
                        $message .= $buf_o . "\n";
                        $buf = $line_part[$e];
                    }
                }
                $message .= $buf . "\n";
            }
            return ($message);
        }
        function create_header() {  //创建邮件头信息
            $this->Start();
            $header = array();
            $header[] = sprintf("Date: %s\r\n", date("D, j M Y G:i:s"));
            $header[] = sprintf("From: %s <%s>\r\n", $this->FromName, trim($
this->From));
            if(count($this->to) > 0)
                $header[] = $this->addr_append("To", $this->to);
            if(count($this->cc) > 0)
                $header[] = $this->addr_append("Cc", $this->cc);
            if(count($this->bcc) > 0)
                $header[] = $this->addr_append("Bcc", $this->bcc);
            if(count($this->ReplyTo) > 0)
                $header[] = $this->addr_append("Reply-to", $this->ReplyTo);
            $header[] = sprintf("Subject: %s\r\n", trim($this->Subject));
            $header[] = sprintf("X-Priority: %d\r\n", $this->Priority);
            $header[] = sprintf("X-Mailer: %s\r\n", $this->Version);
        $header[]=sprintf("Content-Transfer-Encoding:%s\r\n",$this->Encoding);
            $header[] = sprintf("Return-Path: %s\r\n", trim($this->From));
            // 添加自定义头信息
            for($index = 0; $index < count($this->CustomHeader); $index++)
                $header[] = sprintf("%s\r\n", $this->CustomHeader[$index]);
```

```
            if($this->UseMSMailHeaders)
                $header[] = $this->UseMSMailHeaders();
            // 添加所有附件
            if(count($this->attachment) > 0)
            {
                $header[] = sprintf("Content-Type: multipart/mixed; charset=\
"%s\";\r\n", $this->CharSet);
                $header[] = sprintf(" boundary=\"--=%s\"\r\n", $this->boundary);
            }
            else
                $header[] = sprintf("Content-Type: %s; charset=\"%s\";\r\n",
$this->ContentType, $this->CharSet);
            $header[] = "MIME-Version: 1.0\r\n";
            return(join("", $header)."\r\n");
        }
        function create_body() {  //创建信件体
            if($this->WordWrap)
                $this->Body = $this->wordwrap($this->Body, $this->WordWrap);
            if(count($this->attachment) > 0) {
                if(!$body = $this->attach_all())
                    return false;
            }
            else
                $body = $this->Body;
            return(trim($body));
        }
        ////////////////////////////////////////////////
        // 附件
        ////////////////////////////////////////////////
        //添加附件
        function AddAttachment($path, $name = "", $type= "application/octet-stream") {
            if(!@is_file($path)) {
                $this->error_handler(sprintf("Could not find %s file on filesystem",
$path));
                return false;
            }
            $filename = basename($path);
            if($name == "")
                $name = $filename;
            $this->boundary = "_b" . md5(uniqid(time()));
            $cur = count($this->attachment);
            $this->attachment[$cur][0] = $path;
            $this->attachment[$cur][1] = $filename;
            $this->attachment[$cur][2] = $name;
            $this->attachment[$cur][3] = $type;
            return true;
        }
        function attach_all() {  //粘贴所有附件
            $mime = array();
            $mime[] = sprintf("----=%s\n", $this->boundary);
            $mime[] = sprintf("Content-Type: %s\n", $this->ContentType);
            $mime[] = "Content-Transfer-Encoding: 8bit\n\n";
```

```
            $mime[] = sprintf("%s\n", $this->Body);
            for($i = 0; $i < count($this->attachment); $i++) {
                $path = $this->attachment[$i][0];
                $filename = $this->attachment[$i][1];
                $name = $this->attachment[$i][2];
                $type = $this->attachment[$i][3];
                $mime[] = sprintf("----=%s\n", $this->boundary);
                $mime[] = "Content-Type: $type;\n";
                $mime[] = sprintf("name=\"%s\"\n", $name);
                $mime[] = "Content-Transfer-Encoding: base64\n";
                $mime[] = sprintf("Content-Disposition: attachment; filename=\
"%s\"\n\n", $name);
                if(!$mime[] = sprintf("%s\n\n", $this->encode_file($path)))
                    return false;
            }
            $mime[] = sprintf("\n----=%s--\n", $this->boundary);
            return(join("", $mime));
        }
        function encode_file ($path) {  //对附件进行编码
            if(!@$fd = fopen($path, "r"))
            {
                $this->error_handler("File Error: Could not open file $path");
                return false;
            }
            $file = fread($fd, filesize($path));
            $encoded = chunk_split(base64_encode($file));
            fclose($fd);
            return($encoded);
        }
        ////////////////////////////////////////////////
        //清空变量
        ////////////////////////////////////////////////
        //.....省略部分代码
        ////////////////////////////////////////////////
        function error_handler($msg) {  //错误处理
            $this->ErrorAlerts[] = $msg;
            if($this->MailerDebug == true) {
                print("<h3>Mailer Error</h3>");
                print("Description:<br>");
                printf("<font color=\"FF0000\">%s</font>", $msg);
            }
        }
        function AddCustomHeader($custom_header) {  //添加自定义头部
            $this->CustomHeader[] = $custom_header;
        }
        function UseMSMailHeaders() {  //根据邮件的优先级来创建邮件头
            $MSHeader = "";
            if($this->Priority == 1)
                $MSPriority = "High";
            elseif($this->Priority == 5)
                $MSPriority = "Low";
            else
```

```
            $MSPriority = "Medium";
        $MSHeader .= sprintf("X-MSMail-Priority: %s\n", $MSPriority);
        $MSHeader .= sprintf("Importance: %s\n", $MSPriority);
        return($MSHeader);
    }
}
// End of class
class SMTP {  //下面创建一个 SMTP 类
    var $SMTP_PORT = 25; //默认端口
    var $CRLF = "\r\n";  //回车换行符
    var $smtp_conn;     //连接到服务器的 socket
    var $error;        //保存错误信息
    var $helo_rply;     //服务器针对 HELO 命令的响应
    var $do_debug;      //是否为调试模式
    /*
    * SMTP()构造函数，初始化变量
    */
    function SMTP() {
        $this->smtp_conn = 0;
        $this->error = null;
        $this->helo_rply = null;
        $this->do_debug = 0;
    }

    /*
    * Connected()函数
    * 如果连接服务器成功，则返回 true，否则返回 false
    */
    function Connected() {
        if(!empty($this->smtp_conn)) {
            $sock_status = socket_get_status($this->smtp_conn);
            if($sock_status["eof"]) {
# hmm this is an odd situation... the socket is
# valid but we aren't connected anymore
                if($this->do_debug >= 1) {
                    echo "SMTP -> NOTICE:" . $this->CRLF .
                        "EOF caught while checking if connected";
                }
                $this->Close();
                return false;
            }
            return true; # everything looks good
        }
        return false;
    }
    /*
    * 连接服务器函数
    * 如果成功返回: 220
    * 失败: 421
    */
    function Connect($host,$port=0,$tval=30) {
        $this->error = null;
```

```
        if($this->connected()) {  //检查是否已经连接上服务器
            $this->error =
                array("error" => "Already connected to a server");
            return false;
        }
        if(empty($port)) {  //判断端口是否为空，如果为空，则使用默认端口
            $port = $this->SMTP_PORT;
        }
        //调用 fsockopen 函数连接远程服务器
        $this->smtp_conn = fsockopen($host,    //服务器地址
            $port,   //端口号
            $errno,   //错误编码
            $errstr,  //错误内容
            $tval);   //重新链接的时间间隔
        if(empty($this->smtp_conn)) {  //如果失败
            $this->error = array("error" => "Failed to connect to server",
                "errno" => $errno,
                "errstr" => $errstr);
            return false;
        }
        //获取响应
        $announce = $this->get_lines();
        return true;
    }
    /*
    断开连接 */
    function Close() {
        $this->error = null; # so there is no confusion
            $this->helo_rply = null;
        if(!empty($this->smtp_conn)) {
# close the connection and cleanup
            fclose($this->smtp_conn);
            $this->smtp_conn = 0;
        }
    }
    /*************************************************************
    *                    SMTP COMMANDS                          *
    *************************************************************/
    /*
    * Data($msg_data)
    * SMTP CODE INTERMEDIATE: 354
    *    [data]
    *    <CRLF>.<CRLF>
    *    SMTP CODE SUCCESS: 250
    *    SMTP CODE FAILURE: 552,554,451,452
    * SMTP CODE FAILURE: 451,554
    * SMTP CODE ERROR  : 500,501,503,421
    */
    function Data($msg_data) {  //发送邮件内容
        $this->error = null; //清空错误
        if(!$this->connected()) {  //创建连接，如果连接失败
            $this->error = array(
```

```
                "error" => "Called Data() without being connected");
            return false;
        }
        $this->send_line("DATA");  //发送 Data 命令
        $rply = $this->get_lines();  //获取响应码
        $code = substr($rply,0,3);
        if($this->do_debug >= 2) {   //显示状态
            echo "SMTP -> FROM SERVER:" . $this->CRLF . $rply;
        }
        if($code != 354) {  //如果响应码为 354,
            $this->error =
                array("error" => "DATA command not accepted from server",
                "smtp_code" => $code,
                "smtp_msg" => substr($rply,4));
            if($this->do_debug >= 1) {
                echo "SMTP -> ERROR: " . $this->error["error"] .
                    ": " . $rply . $this->CRLF;
            }
            return false;
        }
        $field = substr($lines[0],0,strpos($lines[0],":"));
        $in_headers = false;
        if(!empty($field) && !strstr($field," ")) {
            $in_headers = true;
        }
        $max_line_length = 998; # used below; set here for ease in change

            while(list(,$line) = @each($lines)) {
                $lines_out = null;
                if($line == "" && $in_headers) {
                    $in_headers = false;
                }
                while(strlen($line) > $max_line_length) {
                    $pos = strrpos(substr($line,0,$max_line_length)," ");
                    $lines_out[] = substr($line,0,$pos);
                    $line = substr($line,$pos + 1);
                    if($in_headers) {
                        $line = "\t" . $line;
                    }
                }
                $lines_out[] = $line;

                while(list(,$line_out) = @each($lines_out)) {
                    if($line_out[0] == ".") {
                        $line_out = "." . $line_out;
                    }
                    fputs($this->smtp_conn,$line_out . $this->CRLF);
                }
            }
            $this->send_line($this->CRLF.".");
            $rply = $this->get_lines();
            $code = substr($rply,0,3);
```

```
            if($this->do_debug >= 2) {
                echo "SMTP -> FROM SERVER:" . $this->CRLF . $rply;
            }
            if($code != 250) {
                $this->error =
                    array("error" => "DATA not accepted from server",
                    "smtp_code" => $code,
                    "smtp_msg" => substr($rply,4));
                if($this->do_debug >= 1) {
                    echo "SMTP -> ERROR: " . $this->error["error"] .
                        ": " . $rply . $this->CRLF;
                }
                return false;
            }
            return true;
        }
        function Hello($host="") {  //发送 HELO 命令到服务器，试图与服务器对话
            $this->error = null; # so no confusion is caused
            if(!$this->connected()) {
                $this->error = array(
                    "error" => "Called Hello() without being connected");
                return false;
            }
            if(empty($host)) {
                $host = "localhost";
            }
            $this->send_line("HELO " . $host);
            $rply = $this->get_lines();
            $code = substr($rply,0,3);
            if($this->do_debug >= 2) {
                echo "SMTP -> FROM SERVER:" . $this->CRLF . $rply;
            }
            if($code != 250) {
                $this->error =
                    array("error" => "HELO not accepted from server",
                    "smtp_code" => $code,
                    "smtp_msg" => substr($rply,4));
                if($this->do_debug >= 1) {
                    echo "SMTP -> ERROR: " . $this->error["error"] .
                        ": " . $rply . $this->CRLF;
                }
                return false;
            }
            $this->helo_rply = $rply;

            return true;
        }
        //进行服务器用户名和密码验证
        function AuthHello($host="",$user="",$pass="") {
            $this->error = null; # so no confusion is caused
            if(!$this->connected()) {
                $this->error = array(
```

```
                    "error" => "Called Hello() without being connected");
                return false;
            }
            if(empty($host)) {
                $host = "localhost";
            }
            $this->send_line("EHLO ".$host);
            $rply = $this->get_lines();
            $code = substr($rply,0,3);
            if($code != 250) {
                $this->error =
                    array("error" => "EHLO not accepted from server",
                    "smtp_code" => $code,
                    "smtp_msg" => substr($rply,4));
                return false;
            }
            $this->helo_rply = $rply;
            $this->send_line("AUTH LOGIN");
            $rply = $this->get_lines();
            $code = substr($rply,0,3);
            if($code != 334) {
                $this->error =
                    array("error" => "AUTH LOGIN not accepted from server",
                    "smtp_code" => $code,
                    "smtp_msg" => substr($rply,4));
                return false;
            }
            $this->send_line(base64_encode($user));
            $rply = $this->get_lines();
            $code = substr($rply,0,3);

            if($code != 334) {
                $this->error =
                    array("error" => "USER not accepted from server",
                    "smtp_code" => $code,
                    "smtp_msg" => substr($rply,4));
                return false;
            }
            $this->send_line(base64_encode($pass));
            $rply = $this->get_lines();
            $code = substr($rply,0,3);
            if($code != 235) {
                $this->error =
                    array("error" => "PASSWORD not accepted from server",
                    "smtp_code" => $code,
                    "smtp_msg" => substr($rply,4));
                return false;
            }
            return true;
        }
        function MailFrom($from) {// 发送 MAIL FROM 命令
            $this->error = null; # so no confusion is caused
```

```
            if(!$this->connected()) {
                $this->error = array(
                    "error" => "Called Mail() without being connected");
                return false;
            }
            $this->send_line("MAIL FROM:" . $from);
            //fputs($this->smtp_conn,"MAIL FROM:" . $from . $this->CRLF);
            $rply = $this->get_lines();
            $code = substr($rply,0,3);
            if($this->do_debug >= 2) {
                echo "SMTP -> FROM SERVER:" . $this->CRLF . $rply;
            }
            if($code != 250) {
                $this->error =
                    array("error" => "MAIL not accepted from server",
                    "smtp_code" => $code,
                    "smtp_msg" => substr($rply,4));
                if($this->do_debug >= 1) {
                    echo "SMTP -> ERROR: " . $this->error["error"] .
                        ": " . $rply . $this->CRLF;
                }
                return false;
            }
            return true;
        }
        function Quit($close_on_error=true) {  //发送 QUIT 命令到服务器
            $this->error = null; # so there is no confusion
            if(!$this->connected()) {
                $this->error = array(
                    "error" => "Called Quit() without being connected");
                return false;
            }
            $this->send_line("QUIT");
            //fputs($this->smtp_conn,"quit" . $this->CRLF);
            $byemsg = $this->get_lines();
            if($this->do_debug >= 2) {
                echo "SMTP -> FROM SERVER:" . $this->CRLF . $byemsg;
            }
            $rval = true;
            $e = null;
            $code = substr($byemsg,0,3);
            if($code != 221) {
                $e = array("error" => "SMTP server rejected quit command",
                    "smtp_code" => $code,
                    "smtp_rply" => substr($byemsg,4));
                $rval = false;
                if($this->do_debug >= 1) {
                    echo "SMTP -> ERROR: " . $e["error"] . ": " .
                        $byemsg . $this->CRLF;
                }
            }
```

```
            if(empty($e) || $close_on_error) {
                $this->Close();
            }
            return $rval;
        }
        function Recipient($to) {  //发送 RCPT TO 命令到服务器
            $this->error = null; # so no confusion is caused
            if(!$this->connected()) {
                $this->error = array(
                    "error" => "Called Recipient() without being connected");
                return false;
            }
            $this->send_line("RCPT TO:" . $to);
            //fputs($this->smtp_conn,"RCPT TO:" . $to . $this->CRLF);
            $rply = $this->get_lines();
            $code = substr($rply,0,3);
            if($this->do_debug >= 2) {
                echo "SMTP -> FROM SERVER:" . $this->CRLF . $rply;
            }
            if($code != 250 && $code != 251) {
                $this->error =
                    array("error" => "RCPT not accepted from server",
                    "smtp_code" => $code,
                    "smtp_msg" => substr($rply,4));
                if($this->do_debug >= 1) {
                    echo "SMTP -> ERROR: " . $this->error["error"] .
                        ": " . $rply . $this->CRLF;
                }
                return false;
            }
            return true;
        }
        function get_lines() {  //获取服务器响应
            $data = "";
            while($str = fgets($this->smtp_conn,515)) {
                $data .= $str;
                if(substr($str,3,1) == " ") { break; }
            }
            if($this->do_debug >= 4) {
                echo "SMTP <- get_lines(): \$data is \"$data\"" . $this->CRLF;
            }
            return $data;
        }
        function send_line($data) {  //发送命令
            fputs($this->smtp_conn,$data.$this->CRLF);
            if($this->do_debug >= 4)
                echo "SMTP -> send_line(): \$data is \"$data\"" . $this->CRLF;
        }

    }
    ?>
```

提示　要发送邮件，还需要利用到另外一个类，即 mime_decode 类，该类是将邮件的信息进行 MIME 编码，然后发送。该类保存在文件 class_mime_decode.php 中，由于篇幅有限，这里省略其代码。

以上的文件代码比较复杂，读者需要具备一些基本的 SMTP 协议的知识才容易理解。

19.3.6　创建文件夹

在菜单中单击“文件夹”时，可以查看现有的文件夹，同时可以创建新的文件夹，如图 19-11 所示。

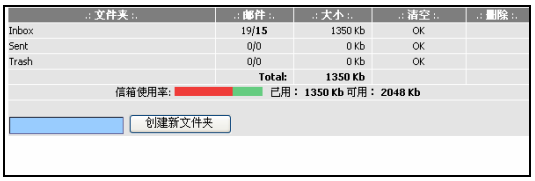


图 19-11　显示并创建文件夹

19.3.7　通讯录管理

在功能菜单中单击“通讯录”可以查看现有的所有通讯录，如图 19-12 所示。通讯录的详细信息如图 19-13 所示。



图 19-12　通讯录列表

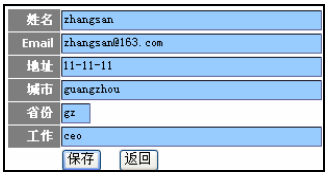


图 19-13　通讯录详细信息

通讯录功能由 addressbook.php 文件实现，代码参见光盘。

19.3.8　系统设置

系统设置的界面如图 19-14 所示。在该界面中，用户可以进行个人信息设置、垃圾箱设置、发邮件设置、收邮件设置以及签名设置等。



图 19-14　系统设置

系统设置由文件 preferences.php 文件实现，代码参见光盘。

第 20 章　在线书店系统

本章将介绍一个功能比较完善的在线书店系统，该系统能够实现书店的基本功能，读者只需要稍加修改，便能得到一个实用的网络书店系统。本系统最大的特点就是管理功能非常强大，如特价图书管理、商场信息发布（如打折信息）、推荐图书管理、热点图书管理和新书管理等。

在线书店系统要实现以下基本功能：

（1）浏览图书详细信息；
（2）搜索图书；
（3）高级搜索；
（4）在线购书；
（5）修改购书数量；
（6）给图书投票；
（7）显示投票结果；
（8）管理购物车；
（9）统计购书金额；
（10）删除定购图书；
（11）安全登录管理；
（12）会员管理；
（13）订单管理；
（14）图书管理；
（15）分类管理；
（16）信息编辑；
（17）信息目录；
（18）信用卡类型管理。

以上是在线书店系统的核心功能，其他许多功能读者可以在后面的代码学习中了解到。



提示　为了让本系统有更多的适应性，光盘中对数据库提供了 3 个版本，即后台数据库分别为 Access、MySQL 以及 SQL Server，读者可以根据需要来选择学习和改进。

20.1　数据库设计

在线书店系统一共由 7 个数据库表组成。下面依次介绍这些数据库表的结构。

items 表用于保存图书详细信息，其设计结构如图 20-1 所示。



items : 表

	字段名称	数据类型	
🔑	item_id	自动编号	
	category_id	数字	图书分类编号
	name	文本	图书名称
	author	文本	图书作者
	price	数字	价格
	product_url	文本	书本链接
	image_url	文本	封面图像链接
	notes	备注	图书说明
	is_recommended	数字	是否被推荐
	rating	数字	投票结果
	rating_count	数字	投票数量

图 20-1　item 表设计结构

categories 表用于保存图书的分类信息。为了便于图书管理，通常会将图书分类，如计算机类、管理类、人文类等。categories 表的设计结构如图 20-2 所示。

members 表用于保存网络书店的会员信息，如会员的姓名、联系方式等。members 表的设计结构如图 20-3 所示。



categories : 表

	字段名称	数据类型	
🔑	category_id	自动编号	图书分类编号
	name	文本	图书分类名称

图 20-2　categories 表设计结构



members : 表

	字段名称	数据类型	
🔑	member_id	自动编号	会员编号
	member_login	文本	会员帐号
	member_password	文本	会员密码
	member_level	数字	会员级别
	first_name	文本	名字
	last_name	文本	姓氏
	email	文本	Email
	phone	文本	联系电话
	address	文本	联系地址
	notes	备注	备注
	card_type_id	数字	卡类型
	card_number	文本	卡号码

图 20-3　members 表设计结构

orders 表用于保存图书订单的信息，如图书的编号、会员编号以及定购数量等。orders 表的设计结构如图 20-4 所示。

card_types 表用于保存卡的类型，通常卡类型有信用卡、普通银行卡等。card_types 表的设计结构如图 20-5 所示。



orders : 表

	字段名称	数据类型	
🔑	order_id	自动编号	订单编号
	member_id	数字	会员编号
	item_id	数字	图书编号
	quantity	数字	定购数量

图 20-4　orders 表设计结构



card_types : 表

	字段名称	数据类型	
🔑	card_type_id	自动编号	卡类型编号
	name	文本	卡类型名称

图 20-5　card_types 表设计结构

editorial_categories 表用于设置各种系统管理信息，如需要增加商店通知信息、打折信息、特价图书管理、最新图书等，都可以通过这个表来设置。editorial_categories 表的设计结构如图 20-6 所示。

editorials 表用于设定 editorial_categories 表中的各种信息分类的内容，如输入商场的每日新闻、推荐图书等信息。editorials 表结构如图 20-7 所示。



editorial_categories : 表

	字段名称	数据类型	
🔑	editorial_cat_id	自动编号	信息编号
	editorial_cat_name	文本	信息名称

图 20-6　editorial_categories 表设计结构



editorials : 表

	字段名称	数据类型	
🔑	article_id	自动编号	
	editorial_cat_id	数字	信息分类编号
	article_title	文本	信息标题
	article_desc	备注	信息内容
	item_id	数字	图书编号

图 20-7　editorials 表设计结构

20.2 在线书店的客户端功能

本系统功能比较完善，因此包含了较多文件。为了便于读者的二次开发，所有的后台 PHP 代码和前台显示的 HTML 代码，都分开作为两个文件来处理。这样大大提高了代码的可读性。

本系统主要包含了以下文件。

（1）Common.php，用于数据库连接设置以及公共函数的声明。

（2）Default.php（Default.html），为系统的首页，用于各种功能导航。

（3）Header.php（Header.html），为页面的头部导航文件，因为每个页面都包含了头部导航部分，因此做成一个通用文件，这样在每个页面都可以使用。

（4）Footer.php（Footer.html），为页面的尾部导航文件，因为每个页面都包含了尾部导航部分，因此做成一个通用文件。

（5）Login.php（Login.html），用于会员或者管理员登录，只有会员才有资格进行在线图书定购。

（6）Registration.php（Registration.html），用于会员注册。

（7）BookDetail.php（BookDetail.html），用于阅读书籍的详细信息。

（8）Books.php（Books.html），用于图书查询。

（9）AdvSearch.php（AdvSearch.html），用于图书的高级搜索。

（10）MyInfo.php（MyInfo.html），用于显示会员的详细信息。

（11）ShoppingCart.php（ShoppingCart.html），用于查看购物车的信息。

（12）ShoppingCartRecord.php（ShoppingCartRecord.html），用于编辑订单的信息，同时也可以删除订单内容。

以上是客户端的功能，下面介绍管理员功能的文件。

（1）AdminMenu.php（AdminMenu.html），该文件为管理员登录主界面，用于管理员功能导航。

（2）MembersGrid.php（MembersGrid.html），用于显示所有的会员信息，并且能够进行会员搜索。

（3）MembersInfo.php（MembersInfo.html），用于显示会员的具体信息，同时能够显示该会员的购物信息。

（4）OrdersGrid.php（OrdersGrid.html），用于显示所有会员的订单信息，并且能够进行订单搜索。

（5）OrdersRecord.php（OrdersRecord.html），用于管理员编辑、删除会员的订单信息。

（6）AdminBooks.php（AdminBooks.html），用于查看所有的图书信息，分页显示记录，进行图书搜索。

（7）BookMaint.php（BookMaint.html），用于管理员新增、编辑或者删除图书信息。

（8）CategoriesGrid.php（CategoriesGrid.html），用于显示所有的图书分类信息。

（9）CategoriesRecord.php（CategoriesRecord.html），用于新增、编辑或者删除分类信息。

（10）EditorialsGrid.php（EditorialsGrid.html），用于显示所有的网站定制信息。

（11）EditorialsRecord.php（EditorialsRecord.html），用于新增、编辑或者删除具体的网站定制信息。

（12）EditorialCatGrid.php（EditorialCatGrid.html），用于显示所有的网站定制信息类型。

（13）EditorialCatRecord.php（EditorialCatRecord.html），用于新增、编辑或者删除网站定制信息类型。

（14）CardTypesGrid.php（CardTypesGrid.html），用于查看所有的卡类型信息。

（15）CardTypesRecord.php（CardTypesRecord.html），用于新增、编辑或者删除卡类型。

同时系统还提供了 3 个用于数据库连接的文件，分别用来连接 ODBC 数据库、MySQL 数据库以及 SQL Server 数据库。

（1）db_odbc.inc，用于连接 ODBC 数据库。

（2）db_mysql.inc，用于连接 MySQL 数据库。

（3）db_mssql.inc，用于连接 SQL Server 数据库。

以上介绍了本系统重要的文件，还有很多文件没有列出来，读者可以参考本书光盘的源代码进行学习。

提示　由于本系统中很多功能模块都是通用的，为了减少代码量，省略了很多重复的代码。

20.2.1　通用文件（Common.php）

Common.php 文件是一个重要的文件，该文件中实现了数据库的连接，以及各种通用函数的声明。其他网页如果需要连接数据库或者使用某些函数，只需要包含该文件即可。由于该文件设计比较合理，因此多次被使用。Common.php 文件如代码 20-1 所示。

【代码 20-1】　Common.php

```
<?php
error_reporting (E_ALL ^ E_NOTICE);
include("./template.php");
// 不同的后台数据库要包含不同的数据库连接文件
//连接 odbc 数据库，则包含 db_odbc.inc 文件
//连接 SQL Server，则包含 db_mssql.inc
//include("./db_odbc.inc");
//include("./db_mssql.inc");
//书中仍以 MySQL 数据库为基础介绍
include("./db_mysql.inc");
define("DATABASE_NAME","bookstore");
define("DATABASE_USER","root");
define("DATABASE_PASSWORD","123456");
define("DATABASE_HOST","localhost");
// 数据库初始化
$db = new DB_Sql();  //创建 db_sql 类
$db->Database = DATABASE_NAME;
$db->User     = DATABASE_USER;
$db->Password = DATABASE_PASSWORD;
$db->Host     = DATABASE_HOST;
$app_path = ".";
$header_filename = "Header.html";  //页面头文件
$footer_filename = "Footer.html";  //页面尾文件
function tohtml($strValue)  //显示 html 实体，即在浏览器中直接显示 html 标记
```

```
    {
        return htmlspecialchars($strValue);
    }
    function tourl($strValue)  //进行 url 编码
    {
        return urlencode($strValue);
    }

    function get_param($param_name)  //获取属性值
    {
        global $HTTP_POST_VARS;
        global $HTTP_GET_VARS;
        $param_value = "";
        if(isset($HTTP_POST_VARS[$param_name]))
            $param_value = $HTTP_POST_VARS[$param_name];
        else if(isset($HTTP_GET_VARS[$param_name]))
            $param_value = $HTTP_GET_VARS[$param_name];
        return $param_value;
    }
    function get_session($param_name)  //获取会话变量值
    {
        global $HTTP_POST_VARS;
        global $HTTP_GET_VARS;
        global ${$param_name};
        $param_value = "";
        if(!isset($HTTP_POST_VARS[$param_name]) && !isset($HTTP_GET_VARS[$param_
name]) && session_is_registered($param_name))
            $param_value = ${$param_name};
        return $param_value;
    }
    function set_session($param_name, $param_value)  //创建会话
    {
        global ${$param_name};
        if(session_is_registered($param_name))
            session_unregister($param_name);
        ${$param_name} = $param_value;
        session_register($param_name);
    }
    function is_number($string_value)  //判断是否为数值
    {
        if(is_numeric($string_value) || !strlen($string_value))
            return true;
        else
            return false;
    }
    function tosql($value, $type)  //将字符串转换成 sql
    {
        if($value == "")
            return "NULL";
        else
            if($type == "Number")  //如果为数值
                return doubleval($value);
```

```
        else
        {   //替换字符串中的转义字符
            if(get_magic_quotes_gpc() == 0)
            {
                $value = str_replace("'","''",$value);
                $value = str_replace("\\","\\\\",$value);
            }
            else
            {
                $value = str_replace("\\'","''",$value);
                $value = str_replace("\\\"","\"",$value);
            }

            return "'" . $value . "'";
        }
}
function strip($value)  //去掉转义字符串中的斜杠
{
    if(get_magic_quotes_gpc() == 0)
        return $value;
    else
        return stripslashes($value);
}
function db_fill_array($sql_query)  //将查询结果放入数组中
{
    $db_fill = new DB_Sql();
    $db_fill->Database = DATABASE_NAME;
    $db_fill->User    = DATABASE_USER;
    $db_fill->Password = DATABASE_PASSWORD;
    $db_fill->Host    = DATABASE_HOST;
    $db_fill->query($sql_query);
    if ($db_fill->next_record())
    {
        do
        {
            $ar_lookup[$db_fill->f(0)] = $db_fill->f(1);
        } while ($db_fill->next_record());
        return $ar_lookup;
    }
    else
        return false;

}
//创建查询
function dlookup($table_name, $field_name, $where_condition)
{
    $db_look = new DB_Sql();
    $db_look->Database = DATABASE_NAME;
    $db_look->User    = DATABASE_USER;
    $db_look->Password = DATABASE_PASSWORD;
    $db_look->Host    = DATABASE_HOST;
    $db_look->query("SELECT " . $field_name . " FROM " . $table_name . " WHERE
```

```
" . $where_condition);
        if($db_look->next_record())
            return $db_look->f(0);
        else
            return "";
    }
    //获取复选框的值
    function get_checkbox_value($value, $checked_value, $unchecked_value, $type)
    {
        if(!strlen($value))
            return tosql($unchecked_value, $type);
        else
            return tosql($checked_value, $type);
    }
    function get_lov_value($value, $array)
    {
        $return_result = "";
        if(sizeof($array) % 2 != 0)
            $array_length = sizeof($array) - 1;
        else
            $array_length = sizeof($array);
        reset($array);
        for($i = 0; $i < $array_length; $i = $i + 2)
        {
            if($value == $array[$i]) $return_result = $array[$i+1];
        }
        return $return_result;
    }
    //检查用户权限
    function check_security($security_level)
    {
        global $UserRights;
        if(!session_is_registered("UserID"))
            header ("Location: Login.php?querystring=" . tourl(getenv("QUERY_
STRING")) . "&ret_page=" . tourl(getenv("REQUEST_URI")));
        else
            if(!session_is_registered("UserRights") || $UserRights < $security_level)
                header ("Location: Login.php?querystring=" . tourl(getenv("QUERY_
STRING")) . "&ret_page=" . tourl(getenv("REQUEST_URI")));
    }
    ?>
```

20.2.2 数据库连接类（db_mysql.inc）

在 common.php 中，根据数据库的不同，需要包含 3 个不同的数据库连接（inc）文件。这 3 个文件都将数据库连接封装在一个类中。类中提供了各种常用的数据库操作函数。下面以 db_mysql.inc 文件为例进行介绍，如代码 20-2 所示。

【代码 20-2】 db_mysql.inc

```
    <?php
    class DB_Sql {
```

```
        /* 数据库连接属性 */
        var $Host    = "";
        var $Database = "";
        var $User    = "";
        var $Password = "";
        /* public: configuration parameters */
        var $Auto_Free    = 0;    ## 设置为 1 表示自动释放结果
            var $Debug        = 0;    ##设置为 1 表示调试模式
            var $Halt_On_Error = "yes"; ## "yes" (halt with message), "no" (ignore
errors quietly), "report" (ignore errror, but spit a warning)
            var $Seq_Table    = "db_sequence";
        var $Record   = array();  //保存结果集
        var $Row;  //保存行号
        var $Errno    = 0;  //错误编号
        var $Error    = "";  //错误内容
        var $type    = "mysql";  //数据库类型
        var $revision = "5.0";  //数据库版本
        var $Link_ID  = 0;  //数据库连接句柄
        var $Query_ID = 0;  //数据库查询句柄
        /* public: constructor */
        function DB_Sql($query = "") {
            $this->query($query);
        }
        /* 获取连接句柄*/
        function link_id() {
            return $this->Link_ID;
        }
        //获取查询句柄
        function query_id() {
            return $this->Query_ID;
        }
        /*连接数据库函数 */
    function connect($Database = "", $Host = "", $User = "", $Password = "") {
            /* Handle defaults */
            if ("" == $Database)
                $Database = $this->Database;
            if ("" == $Host)
                $Host    = $this->Host;
            if ("" == $User)
                $User    = $this->User;
            if ("" == $Password)
                $Password = $this->Password;
            /* 建立连接，选择数据库*/
            if ( 0 == $this->Link_ID ) {
                $this->Link_ID=mysql_pconnect($Host, $User, $Password);
                if (!$this->Link_ID) {
                    $this->halt("pconnect($Host, $User, \$Password) failed.");
                    return 0;
                }
                if (!mysql_select_db($Database,$this->Link_ID)) {
                    $this->halt("cannot use database ".$this->Database);
                    return 0;
```

```
            }
        }
        return $this->Link_ID;
    }
    /*释放查询结果 */
    function free() {
        mysql_free_result($this->Query_ID);
        $this->Query_ID = 0;
    }
    /* 创建一个查询*/
    function query($Query_String) {
        /*如果为空*/
        if ($Query_String == "")
        return 0;  //返回
        if (!$this->connect()) {
            return 0; /*如果数据库没有连接，返回 */
        };
        # 如果之前有查询，则释放
        if ($this->Query_ID) {
            $this->free();
        }
        if ($this->Debug)  //如果是调试模式，则显示中间结果
            printf("Debug: query = %s<br>\n", $Query_String);
        $this->Query_ID = mysql_query($Query_String,$this->Link_ID);
        $this->Row   = 0;
        $this->Errno = mysql_errno();
        $this->Error = mysql_error();
        if (!$this->Query_ID) {
            $this->halt("Invalid SQL: ".$Query_String);
        }
        # 返回查询句柄
        return $this->Query_ID;
    }
    /* 查询下一条记录 */
    function next_record() {
        if (!$this->Query_ID) {
            $this->halt("next_record called with no query pending.");
            return 0;
        }
        $this->Record = mysql_fetch_array($this->Query_ID);
        $this->Row   += 1;
        $this->Errno  = mysql_errno();
        $this->Error  = mysql_error();
        $stat = is_array($this->Record);
        if (!$stat && $this->Auto_Free) {
            $this->free();
        }
        return $stat;
    }
    /* 定位数据库记录*/
    function seek($pos = 0) {
        $status = mysql_data_seek($this->Query_ID, $pos);
```

```
        if ($status)
            $this->Row = $pos;
        else {
$this->halt("seek($pos) failed: result has ".$this->num_rows()." rows");
            mysql_data_seek($this->Query_ID, $this->num_rows());
            $this->Row = $this->num_rows;
            return 0;
        }
        return 1;
    }
    /* 锁住表 */
    function lock($table, $mode="write") {
        $this->connect();
        $query="lock tables ";
        if (is_array($table)) {
            while (list($key,$value)=each($table)) {
                if ($key=="read" && $key!=0) {
                    $query.="$value read, ";
                } else {
                    $query.="$value $mode, ";
                }
            }
            $query=substr($query,0,-2);
        } else {
            $query.="$table $mode";
        }
        $res = mysql_query($query, $this->Link_ID);
        if (!$res) {
            $this->halt("lock($table, $mode) failed.");
            return 0;
        }
        return $res;
    }
    function unlock() {  //解锁数据库表
        $this->connect();
        $res = mysql_query("unlock tables");
        if (!$res) {
            $this->halt("unlock() failed.");
            return 0;
        }
        return $res;
    }
    /*获取查询语句影响的记录数 */
    function affected_rows() {
        return mysql_affected_rows($this->Link_ID);
    }
    function num_rows() {  //当前记录的记录数
        return mysql_num_rows($this->Query_ID);
    }
    function num_fields() {  //字段数
        return mysql_num_fields($this->Query_ID);
    }
```

```
//……省略部分代码
/* 获取数据库表字段的各种属性 */
function metadata($table='',$full=false) {
    $count = 0;
    $id    = 0;
    $res   = array();
    if ($table) {
        $this->connect();
        $id = mysql_list_fields($this->Database, $table);
        if (!$id)
            $this->halt("Metadata query failed.");
    } else {
        $id = $this->Query_ID;
        if (!$id)
            $this->halt("No query specified.");
    }
    $count = mysql_num_fields($id);
    if (!$full) {
        for ($i=0; $i<$count; $i++) {
            $res[$i]["table"] = mysql_field_table ($id, $i);
            $res[$i]["name"]  = mysql_field_name  ($id, $i);
            $res[$i]["type"]  = mysql_field_type  ($id, $i);
            $res[$i]["len"]   = mysql_field_len   ($id, $i);
            $res[$i]["flags"] = mysql_field_flags ($id, $i);
        }
    } else { // full
        $res["num_fields"]= $count;
        for ($i=0; $i<$count; $i++) {
            $res[$i]["table"] = mysql_field_table ($id, $i);
            $res[$i]["name"]  = mysql_field_name  ($id, $i);
            $res[$i]["type"]  = mysql_field_type  ($id, $i);
            $res[$i]["len"]   = mysql_field_len   ($id, $i);
            $res[$i]["flags"] = mysql_field_flags ($id, $i);
            $res["meta"][$res[$i]["name"]] = $i;
        }
    }
    // free the result only if we were called on a table
    if ($table) mysql_free_result($id);
    return $res;
}
/* 错误处理*/
function halt($msg) {
    $this->Error = mysql_error($this->Link_ID);
    $this->Errno = mysql_errno($this->Link_ID);
    if ($this->Halt_On_Error == "no")
        return;
    $this->haltmsg($msg);
    if ($this->Halt_On_Error != "report")
        die("Session halted.");
}
function haltmsg($msg) {  //停止执行
    printf("</td></tr></table><b>Database error:</b> %s<br>\n", $msg);
```

```
            printf("<b>MySQL Error</b>: %s (%s)<br>\n",
                $this->Errno,
                $this->Error);
        }
        function table_names() {  //获取数据库的表名
            $this->query("SHOW TABLES");
            $i=0;
            while ($info=mysql_fetch_row($this->Query_ID))
            {
                $return[$i]["table_name"]= $info[0];
                $return[$i]["tablespace_name"]=$this->Database;
                $return[$i]["database"]=$this->Database;
                $i++;
            }
            return $return;
        }
    }
    ?>
```

20.2.3　主界面文件（Default.php）

Default.php 文件是系统的主界面，当用户在浏览器中输入“http://localhost/bookstore_php/Default.php”时，便会进入在线书店系统，主界面如图 20-8 所示。



图 20-8　在线书店主界面

主界面中显示了很多的内容，如商场打折信息、推荐图书信息、最新图书信息、特价图书信息、重点推荐图书信息和图书搜索等。本系统全部采用了后台和前台代码分离的技术，由于篇幅限制，大部分模块都只介绍后台 PHP 代码的实现。

下面来看 default.php 文件的代码，如代码 20-3 所示。

【代码 20-3】 Default.php

```
<?php
include ("./common.php");
include ("./Header.php");
include ("./Footer.php");
session_start();
$filename = "Default.php";
//创建一个模板类实例
$tpl = new Template($app_path);
//载入文件 Default.html，该文件为纯 html 表单显示文件
$tpl->load_file("Default.html", "main");
//载入页面头文件
$tpl->load_file($header_filename, "Header");
//载入页面为文件
$tpl->load_file($footer_filename, "Footer");
//设置文件名为 default.php
$tpl->set_var("FileName", $filename);
//以下调用各个函数显示整个页面
Menu_show();  //显示功能菜单
Footer_show();
Search_show();  //显示搜索界面
AdvMenu_show();  //显示高级搜索界面
Recommended_show();  //显示推荐图书
What_show();  //显示热门图书
Categories_show(); //显示图书大类
New_show();  //显示新书推荐
Weekly_show();  //显示每周特价图书
Specials_show();  //显示商场各种优惠信息
$tpl->parse("Header", false);
$tpl->parse("Footer", false);
$tpl->pparse("main", false);
//*********************************************************************
//显示查询页面
function Search_show()
{
    global $db;
    global $tpl;
    $tpl->set_var("ActionPage", "Books.php");
    // Set variables with search parameters
    $fldcategory_id = strip(get_param("category_id"));
    $fldname = strip(get_param("name"));
    // 显示字段
    $tpl->set_var("LBcategory_id", "");
    $tpl->set_var("ID", "");
    $tpl->set_var("Value", "All");
    $tpl->parse("LBcategory_id", true);
    $dbcategory_id = new DB_Sql();
    $dbcategory_id->Database = DATABASE_NAME;
    $dbcategory_id->User    = DATABASE_USER;
    $dbcategory_id->Password = DATABASE_PASSWORD;
```

```
        $dbcategory_id->Host     = DATABASE_HOST;
    $dbcategory_id->query("select category_id, name from categories order by 2");
        while($dbcategory_id->next_record())
        {
            $tpl->set_var("ID", $dbcategory_id->f(0));
            $tpl->set_var("Value", $dbcategory_id->f(1));
            if($dbcategory_id->f(0) == $fldcategory_id)
                $tpl->set_var("Selected", "SELECTED" );
            else
                $tpl->set_var("Selected", "");
            $tpl->parse("LBcategory_id", true);
        }
        $tpl->set_var("name", tohtml($fldname));
        $tpl->parse("FormSearch", false);
    }
    //省略其他函数代码
    ?>
```

20.2.4　会员登录文件（Login.php）

Login.php 文件用于管理员或者会员进行登录。当在主界面上选择任何一个图书链接，查看详细信息的时候，会弹出登录页面。同时如果在页面头部单击“Sign In”链接，也可以进入登录页面。登录成功后会自动进入购物车显示界面。登录界面如图 20-9 所示。

如果用户已经登录，单击“Sign In”链接后，则会显示已经登录，用户可以注销登录，如图 20-10 所示。



图 20-9　登录界面



图 20-10　注销登录

在图 20-10 所示中，单击“Logout”按钮，可以注销登录。下面学习 Login.php 文件和 login.html 文件如何将表单显示和后台代码分离。login.html 的文件如代码 20-4 所示。

【代码 20-4】　Login.html 文件

```
    <html>
    <head>
    <title>Book Store</title>
    <meta name="GENERATOR" content="YesSoftware CodeCharge v.1.1.19 / Templates.
ccp build 3/2/2001">
    <meta http-equiv="pragma" content="no-cache">
    <meta http-equiv="expires" content="0">
```

```
    <meta http-equiv="cache-control" content="no-cache">
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
    </head>
    <body style="background-color: #FFFFFF; color: #000000; font-family: Arial,
Tahoma, Verdana, Helveticabackground-color: #FFFFFF; color: #000000; font-family:
Arial, Tahoma, Verdana, Helvetica">
    {Header}<center>
     <table>
      <tr>
       <td valign="top">
       <!--BeginFormLogin-->
        <form action="{FileName}" method="POST">
         <table style="">
          <tr>
           <td style="background-color: #336699; text-align: Center; border-style:
outset; border-width: 1" colspan="2">
            <font style="font-size: 12pt; color: #FFFFFF; font-weight: bold">输
入用户名和密码</font>
          </td>
         </tr>
         <!--BeginLoginError-->
         <tr>
          <td colspan="2" style="background-color: #FFFFFF; border-width: 1">
           <font style="font-size: 10pt; color: #000000">{sLoginErr}</font>
          </td>
         </tr>
         <!--EndLoginError-->
         <input type="hidden" name="ret_page" value="{ret_page}"><input
type="hidden" name="querystring" value="{querystring}">
         <input type="hidden" name="FormName" value="Login">
         <!--BeginUserInd-->
         <tr>
          <td style="background-color: #FFFFFF; border-width: 1">
           <font style="font-size: 10pt; color: #000000">{UserID}</font>
           <input type="hidden" name="FormAction" value="logout">
           <input type="submit" value="Logout">
          </td>
         </tr>
         <!--EndUserInd-->
         <!--BeginLoginAct-->
         <tr>
          <td style="background-color: #FFEAC5; border-style: inset;
border-width: 0">
           <font style="font-size: 10pt; color: #000000">用户名: </font>
          </td>
          <td style="background-color: #FFFFFF; border-width: 1">
           <input type="text" name="Login" value="{Login}" maxlength="50">
          </td>
         </tr>
         <tr>
          <td style="background-color: #FFEAC5; border-style: inset;
border-width: 0">
```

```
          <font style="font-size: 10pt; color: #000000">密码：</font>
         </td>
         <td style="background-color: #FFFFFF; border-width: 1">
          <input type="password" name="Password" maxlength="50">
         </td>
        </tr>
        <tr>
         <td colspan="2">
          <input type="hidden" name="FormAction" value="login">
          <input type="submit" value="Login">
         </td>
        </tr>
        <!--EndLoginAct-->
       </table>
      </form>
   <!--EndFormLogin-->
      guest/guest<br>
   admin/admin
     </td>
    </tr>
   </table>
   {Footer}
   <center><font face="Arial"><small> <a href="http://forway.zj.com">四维科技
</a></small></font></center>
   </body>
   </html>
```

在 Login.htm 文件中，要注意一些关键的字段，如“ <!--BeginFormLogin-->”、“<!—BeginLoginError--> ”、“ <!--EndLoginError--> ”、“ <!--BeginUserInd--> ”、“ <!--EndUserInd--> ”、“<!--BeginLoginAct-->”和“<!--EndLoginAct-->”等。这些标记非常重要，因为会在 login.php 文件中被调用，然后显示出对应的 HTML 表单。如“<!--BeginLoginAct-->”、“<!--EndLoginAct-->”两个标记之间显示的就是让用户输入用户名和密码的表单界面。而 login.php 通过创建类 Template 来调用和处理以上定义的代码块，根据需要输出对应的内容。这样就实现了 HTML 和 PHP 代码的分离。

login.php 的文件代码如代码 20-5 所示。

【代码 20-5】　login.php

```
<?php
include ("./common.php");
include ("./Header.php");
include ("./Footer.php");
session_start();   //启动会话
$filename = "Login.php";
//创建类
$tpl = new Template($app_path);
//载入 login.html，显示在中间区域
$tpl->load_file("Login.html", "main");
//载入页面头文件
$tpl->load_file($header_filename, "Header");
//载入页面尾文件
```

```
    $tpl->load_file($footer_filename, "Footer");
    $tpl->set_var("FileName", $filename);
    $sLoginErr = "";
    //获取提交的参数
    $sAction = get_param("FormAction");
    $sForm = get_param("FormName");  //表单 form 的名称
    switch ($sForm) {
    case "Login":  //如果为 login
        Login_action($sAction);  //调用 Login_action 函数
        break;
    }
    //显示 html 表单
    Menu_show();  //显示菜单
    Footer_show();  //显示页面脚
    Login_show();  //显示页面尾
    $tpl->parse("Header", false);
    $tpl->parse("Footer", false);
    $tpl->pparse("main", false);
    //*************************************************************************
    function Login_action($sAction)  //根据用户提交进行不同的处理
    {
        global $db;
        global $tpl;
        global $sLoginErr;
        switch(strtolower($sAction))
        {
        case "login":  //获取提交按钮的名称
            // Login action
            $sLogin = get_param("Login");  //获取登录名
            $sPassword = get_param("Password");  //获取密码
            //从数据库表中查询记录
            $db->query("SELECT member_id,member_level FROM members WHERE member_
login =" . tosql($sLogin, "Text") . " AND member_password=" . tosql($sPassword,
"Text"));
            if($db->next_record()) //如果有记录
            {
                // 将用户名和密码保存在 session 变量中
                set_session("UserID", $db->f("member_id"));
                set_session("UserRights", $db->f("member_level"));
                $sPage = get_param("ret_page");
                if (strlen($sPage))  //判断当前的页码
                    header("Location: " . $sPage);  //如果不为空定位到指定页面
                else  //否则进入购物车，查看登录用户的订单信息
                    header("Location: ShoppingCart.php");
            }
            else  //用户名密码错误
            {
                $sLoginErr = "Login or Password is incorrect.";
            }

            $tpl->parse("FormLogin", false);
```

```
            break;
        case "logout":
            // 如果名称为 logout，表示退出登录
            //注销 session 变量
            session_unregister("UserID");
            session_unregister("UserRights");
            $tpl->parse("FormLogin", false);
            break;
        }
    }
    function Login_show()  //显示登录窗体
    {
        global $tpl;
        global $sLoginErr;
        global $db;
        $tpl->set_var("sLoginErr", $sLoginErr);
        $tpl->set_var("querystring", get_param("querystring"));
        $tpl->set_var("ret_page", get_param("ret_page"));
        //获取用户名
        if(get_session("UserID") == "")
        {
            // 没有登录
            $tpl->set_var("LogoutAct", "");
            $tpl->set_var("UserInd", "");
            $tpl->set_var("Login", strip(tohtml(get_param("Login"))));
            if($sLoginErr == "")
                $tpl->set_var("LoginError", "");
            else
            {
                $tpl->set_var("sLoginErr", $sLoginErr);
                $tpl->parse("LoginError", false);
            }
            $tpl->parse("LoginAct", false);
        }
        else
        {
            // 用户已经登录
            $db->query("SELECT member_login FROM members WHERE member_id=".
get_session("UserID"));
            $db->next_record();
            $tpl->set_var("LoginError", "");
            $tpl->set_var("LoginAct", "");
            $tpl->set_var("UserID", $db->f("member_login"));
            $tpl->parse("UserInd", false);
        }
        $tpl->parse("FormLogin", false);
    }
    ?>
```

如果用户没有注册，可以单击主页面上的“注册”或者“Registration”链接进入注册页面，如图 20-11 所示。

由于篇幅限制，注册页面的代码省略，主页面文件名为 Registration.php。



图 20-11　会员注册页面

20.2.5　显示图书详细信息（BookDetail.php）

BookDetail.php 文件用于显示图书的详细信息，同时用户可以将商品添加到购物车，也可以对该图书进行评论以及查看投票结果。图书详细信息显示界面如图 20-12 所示。

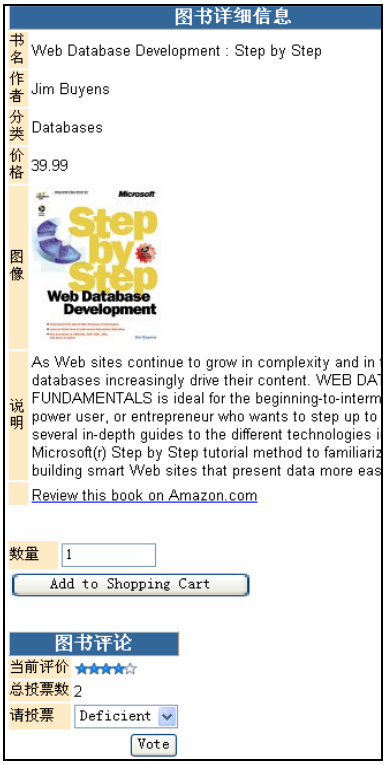


图 20-12　图书详细信息显示

在图 20-12 中，在“数量”中输入需要定购的数量，然后单击“Add to Shopping Cart”，可以将该图书添加到购物车。同时还可以查看读者对本书的评论。评论共分成 5 个级别，用户可以在“选择投票等级”下拉列表框中选择投票等级。

20.2.6　购物篮管理（ShoppingCart.php）

用户登录成功后，将图书添加到购物篮以及在页面头部单击“Shoping Cart”链接，都会

进入购物篮管理页面，如图 20-13 所示。会员只能管理自己的购物篮信息。



图 20-13　购物篮管理页面

在该界面中，可以单击“Details”链接修改、删除定购的信息，同时该页面能够统计定购书籍的总价值，以及显示会员的详细信息。

20.2.7　修改购物信息（ShoppingCartRecord.php）

当用户在购物篮里面单击具体的商品链接时，会进入到该页面。该页面允许用户删除或修改某种商品的定购数量。其运行界面如图 20-14 所示。



图 20-14　修改图书定购信息

在图 20-14 中，输入新的数量后，单击“Update”按钮，可以生成新的定购数量。单击“Delete”按钮能够删除该种图书的定购。

20.2.8　图书查询（Books.php）

本系统提供了非常强大的查询工具。如果只进行普通查询，需要在主界面的查询文本框中输入查询关键字，如图 20-15 所示，单击“Search”按钮，进行查询。



图 20-15　简单搜索

查询后进入“Books.php”页面，结果如图 20-16 所示。

查询结果中会显示所有符合条件的图书数量，同时显示每本书的信息。如果单击书名链接，可以查看每本书的详细信息。

除了简单查询之外，本系统还提供了高级查询。在主界面或者图 20-16 中，单击“高级查询”链接，可以进入高级查询页面（文件名为 AdvSearch.php），如图 20-17 所示。

在高级查询中，除了能够对书名进行查询，还可以对作者、分类、定价进行查询，读者还可以进行扩展，如对日期进行查询等。查询结果同样是在 Books.php 文件中显示出来。

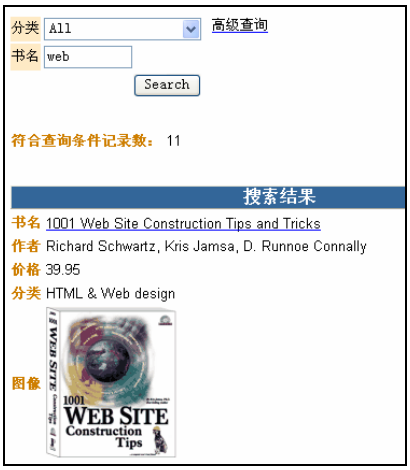


图 20-16　查询结果



图 20-17　高级查询

AdvSearch.php 文件仅仅是一个表单文件，实际提交的数据处理是在 books.php 中进行的。

20.3　在线书店的管理功能

强大的书店管理功能能够给书店的维护人员带来极大的方便。本系统提供了较全面的管理功能，而且功能都采用了模块化的管理，读者只需要搞清楚一个管理功能的实现，便能够方便地开发和扩展其他的功能。

用户在登录的时候以管理员身份登录，则可以进入到管理员界面。本系统默认管理员的账号和密码都为“admin”。登录成功后，单击页面中的“管理员”链接或者“Administration”链接，可以进入到管理员页面，如图 20-18 所示。

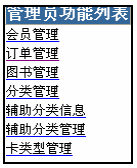


图 20-18　管理员页面

管理员主界面文件名为 AdminMenu.php，该页面主要起导航作用。

20.3.1　会员管理

在管理员菜单中单击“会员管理”即可进入会员管理界面。会员管理文件名为 MembersGrid.php。会员管理界面如图 20-19 所示。



图 20-19　会员管理

在会员管理界面中，如果单击每个具体会员的链接，如“admin”，可以查看该会员的具体信息。另外在会员管理管理中还可以新增会员，同时还增加了方便的搜索功能。

20.3.2　订单管理

在管理员界面中，单击“订单管理”可以进入订单管理的界面。订单管理主要管理所有会员的购书信息，其界面如图 20-20 所示。

在订单管理中，可以根据会员名称来管理，也可以根据书名来管理。另外管理员可以增加订单内容。

订单管理的文件名为 OrdersGrid.php。

在订单管理中，如果单击“添加”链接，可以添加新的订单信息。单击“Edit”链接可以编辑、删除已经存在的订单信息，如图 20-21 所示。



图 20-20　订单管理



图 20-21　新增、编辑和删除订单

在图 20-21 中，单击“Update”可以保存新订单信息或者修改订单信息，单击“Delete”可以删除订单信息，单击“Cancel”可以取消编辑，回到订单管理页面。

编辑订单页面的文件名为 OrdersRecord.php。

20.3.3　图书管理

在管理员界面中，单击“图书管理”链接，能够进入图书管理界面，该页面文件名为 AdminBooks.php。在该页面中，能够查询图书，分页显示所有图书，同时可以添加和编辑新的图书。图书管理的界面如图 20-22 所示。

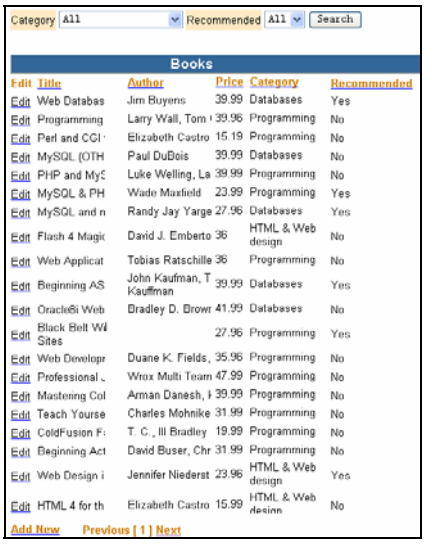


图 20-22　图书管理界面

在图 20-22 中，单击“添加图书”链接可以增加新图书，单击“Edit”链接，可以编辑、删除已经有的图书信息，该页面文件名为 BookMaint.php。

编辑图书信息的界面如图 20-23 所示。

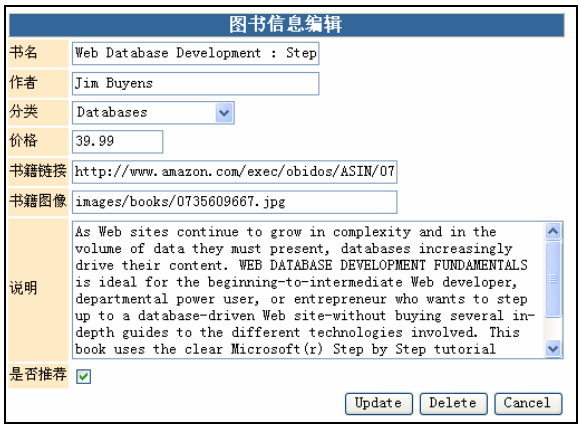


图 20-23　图书编辑界面

20.3.4　分类管理

在管理员界面中，单击“分类管理”，能够进入图书分类管理界面，该页面的文件名为 CategoriesGrid.php。在该页面中能够查看所有的图书分类。同时能够添加或者编辑图书分类。分类管理界面如图 20-24 所示。

在图 20-24 中，单击“分类”链接可以编辑和删除分类，该页面的文件名为 Categories-Record.php。同时可以单击“添加”链接新增图书分类。

图书分类编辑的界面如图 20-25 所示。



图 20-24　图书分类管理界面



图 20-25　图书分类编辑界面

卡类型管理界面如图 20-26 所示。

卡类型编辑界面如图 20-27 所示。



图 20-26　卡类型管理界面



图 20-27　卡类型编辑界面

其他管理分类界面如图 20-28 所示。

其他管理分类编辑界面如图 20-29 所示。

其他信息内容列表界面如图 20-30 所示。



图 20-28　其他管理分类界面



图 20-29　其他管理分类编辑界面



图 20-30　其他信息内容列表界面

其他管理信息内容编辑界面如图 20-31 所示。

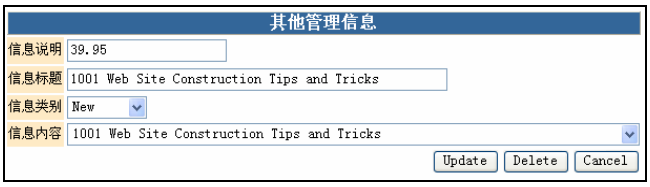


图 20-31　其他管理信息内容编辑界面

本章介绍了一个功能比较完善的在线书店系统，通过在线书店系统，用户能够方便地查询各种图书、定购图书以及统计定购图书的总量。另外本系统提供了强大的管理功能，能够进行各种管理。

因为整个系统的代码量非常庞大，所以读者应结合光盘代码慢慢理解和吸收。